UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

TRABAJO FIN DE GRADO

Sistema basado en web para la realización de prácticas de laboratorio a distancia

María Murillo Moya
Tutor: Alberto Sánchez González
Ponente: Ángel de Castro Martín

JULIO 2015

SISTEMA BASADO EN WEB PARA LA REALIZACIÓN DE PRÁCTICAS DE LABORATORIO A DISTANCIA

Autora: María Murillo Moya

Tutor: Alberto Sánchez González

Ponente: Ángel de Castro Martín

Trabajo realizado en el grupo

Hardware and Control Technology Laboratory

Escuela Politécnica Superior

Universidad Autónoma de Madrid

Julio 2015

Agradecimientos

Llegando al final de este año no puedo olvidarme de agradecer a todas las personas que empezaron, continuaron o han terminado a mi lado en esta gran etapa de mi vida.

En primer lugar, a mi tutor Alberto, por la ayuda prestada, tiempo y dedicación para poder cerrar este Trabajo Fin de Grado a tiempo. Gracias por darme la oportunidad de haberlo realizado y pertenecer al HCTLab.

Cómo no mencionar a mis amigos/as de Tarancón y alrededores. Hacen que cada vez que regreso allí desconecte de la rutina de la universidad pasando muy buenas tardes de risas y noches interminables.

También mención especial al año que viví en Milán, al "*secondo piano"* de la residencia *CdS* por ser como una familia y a todas las personas allí conocidas. A pesar del tiempo transcurrido y distancia que nos separa, es grato no haber perdido el contacto.

Tampoco puedo olvidarme de mis compañeros de universidad, por los buenos momentos dentro y fuera de la EPS. Mención especial a Miriam, Ester, Raquel, David, Álvaro, Rubén, Carlos, Sandra, Laura, Edu, Ryan, Alberto,… Sin todos vosotros habría sido muy difícil llegar hasta aquí.

Por último agradecer infinitamente a mis padres, razón de mi existir. Por apoyarme en mis decisiones, animarme y recordarme que todo esfuerzo vale la pena. Gracias por quererme y ser únicos. También a mi hermana Laura y hermano Alexis, sois esenciales en mi vida y me siento orgullosa de que seáis un ejemplo para mí. No me olvido de mis abuelos y mi tío Eduardo, cualquier éxito os pertenece por haber formado parte de mi vida. Así como al que se ha convertido en el más pequeño de la casa, Darío, por alegrarme con sus sonrisas cada día y poder escuchar sus primeras palabras mientras escribía estas líneas.

María Murillo Moya

Julio 2015

RESUMEN

El control digital aplicado a circuitos electrónicos ha experimentado una gran evolución en las últimas décadas. El diseño de un circuito electrónico se puede dividir en dos etapas claramente diferenciadas. Por una parte, se debe diseñar el propio circuito electrónico (también llamado planta) a controlar, junto al sensado en el caso de implementaciones en lazo cerrado, y por otra parte debe realizarse un regulador que controle dicho circuito.

Con la intención de aislar las posibles fuentes de error es recomendable separar las dos etapas y poder realizar un regulador comprobando su funcionamiento con una planta previamente construida y, posteriormente, realizar el nuevo diseño real de la planta.

En la asignatura Sistemas de Control (3er curso del Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación) las prácticas del laboratorio están basadas en la implementación de un sistema de control digital. Este Trabajo Fin de Grado implementa un laboratorio remoto donde se provee al alumno un circuito electrónico analógico conectado a un dispositivo reprogramable (*Field Programmable Gate Array*) donde los alumnos podrán implementar y probar su regulador digital. El sistema basado en web establecerá la comunicación entre alumno y FPGA utilizando un servidor web conectado a una FPGA, a través de un puerto serie específicamente configurado para la transmisión y recepción de datos entre ambos, mediante una conversión USB-UART (*Universal Asynchronous Receiver-Transmitter)*. A su vez, permitirá la lectura del voltaje de la planta (variable que se desea controlar) y la asignación de los datos al controlador sintetizado en la FPGA de manera telemática, pero con el mismo resultado que si esa asignación se realizase manualmente en el laboratorio.

Para la visualización del resultado se ha integrado un osciloscopio conectado vía LAN al ordenador que actúa de servidor web, monitorizando mediante una sonda la planta del sistema. El resultado podrá ser visualizado en la interfaz web por los alumnos tras ser autenticados y autorizados para el uso de la aplicación por un tiempo limitado mediante un sistema de reservas y usuarios gestionado por una base de datos.

ABSTRACT

Digital control applied to electronic circuits has greatly evolved in the last decades. The design of an electronic circuit can be clearly split in two different stages. On the one hand, the electronic circuit to be controlled (also called plant) must be designed along with the sensing process in case of closed loop implementations. On the other hand, a regulator must be designed to control this circuit.

In order to isolate potential error sources it is recommended to split both stages and develop a regulator checking its behavior with a plant which has been previously built and thereafter develop a new real design of the plant.

In Control Systems course of the 3rd year of the Degree in Telecommunication Technology and Service Engineering, the laboratory practices consist on the implementation of a digital control system for an electronic circuit. This project develops a remote laboratory for the students, where an analog electronic circuit is connected to a reprogrammable device (Field Programmable Gate Array) allowing the students testing and implementing their digital regulators. The web-based system will establish communication between the student and FPGA using a web server connected to a FPGA through a serial port specifically configured for data transmission and reception between them, using an USB-UART converter (Universal Asynchronous Receiver-Transmitter). At the same time, this approach will let reading access to the voltage plant (variable that we want to control) and data assignment to the controller synthesized on the FPGA telematically but with same result as this assignment would be made it manually in the laboratory.

In order to facilitate the visualization of the results, an oscilloscope has been integrated and connected via LAN to the server and monitoring the system plant with a probe. The result could be visualized on the web interface by the students after being successfully authenticated and authorized to use the application for a limited period of time, using a booking system handled within a database.

PALABRAS CLAVE

Sistema basado en web, laboratorio remoto, control digital, *Field Programmable Gate Array*, telemática, *Universal Asynchronous Receiver-Transmitter*, osciloscopio, base de datos.

KEYWORDS

Web-based system, remote laboratory, digital control, *Field Programmable Gate Array*, telematic, *Universal Asynchronous Receiver-Transmitter*, oscilloscope, database.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

GLOSARIO

ADC	Analog to Digital Converter
AJAX	Asynchronous JavaScript and XML
CDN	Content Delivery Network
CLB	Configurable Logic Block
CSS	Cascading Style Sheets
DAC	Digital to Analog Converter
FPGA	Field Programmable Gate Array
HDL	Hardware Language Description
HTML	HyperText Markup Language
IDE	Integrated Developmet Environment
ISE	Integrated Software Environment
JS	JavaScript
PHP	Hypertext Pre-Procesor
PWM	Pulse Width Modulation
SO	Sistema Operativo
SQL	Structured Query Language
TIC	Tecnologías de la Información y Comunicación
TFG	Trabajo Fin de Grado
UART	Universal Asynchronous Receiver/Transmitter
UCF	User Constraint File
URL	Uniform Resource Locator
VHDL	VHSIC Hardware Description Language
XHTML	eXtensible HyperText Markup Language

1. INTRODUCCIÓN

Las prácticas en los laboratorios de los estudios de ingeniería conllevan una importante carga de trabajo por parte de alumnos y profesores. El desarrollo de las Tecnologías de la Información y Comunicación (TIC) en los últimos años ha permitido contribuir a un mejor acceso a sistemas de aprendizaje-educación e interacción con equipos remotos en tiempo real a individuos de cualquier condición [1].

Así surge el desarrollo de los laboratorios convencionales, pasando por los virtuales y alcanzando los remotos [2]. Este último nos permite la interactividad con equipamiento real del laboratorio como pueden ser osciloscopios, fuentes de alimentación, elementos de sensado y actuadores, dispositivos programables digitales, etc., en lugar de programas que simulen los procesos como es el caso de los laboratorios virtuales [3 , 4 , 5]. En la Figura 1 se pueden observar algunas características de ambos.

		Ubicación del alumno	**Dispositivos**
T I P O	**Convencional**	Presencial	Reales o simulados
	Virtual	Remota	Simulado
	Remoto	Remota	Reales

Figura 1.Clasificación de entornos de laboratorio

Este TFG se centra en la realización de un laboratorio remoto sobre reguladores digitales, en la asignatura Sistemas de Control del 3er curso del Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación de la Universidad Autónoma de Madrid. En esta asignatura los alumnos deben implementar un regulador digital para controlar un circuito electrónico analógico, regulando su tensión de salida a través de una actuación basada en PWM (*Pulse-Width Modulation*). El trabajo que deben realizar los alumnos se puede dividir en una etapa de construcción del circuito electrónico y sensado y en otra de diseño del regulador. El sensado se realiza sobre una planta analógica de la cual se quiere controlar su tensión de salida y estará formada por un sencillo circuito de resistencia y condensador (RC) o resistencia, condensador y bobina (RLC). El propósito es que a través de la manipulación de unas variables de control, se pueda regular la tensión de salida de la planta para que alcance unos valores determinados por una consigna. El sistema puede tener modo de funcionamiento en lazo cerrado o abierto, según se compare o no la tensión de

salida del sistema con la consigna, siendo el lazo cerrado más adecuado al poder corregir la actuación del regulador y minimizando así el error.

El módulo de control digital que deben generar los alumnos para el laboratorio de esta asignatura se compone a su vez de otros tres:

- Gestor PWM: se encarga de generar una señal PWM necesaria para el circuito del sistema, que actuará como DAC (*Digital to Analog Converter*). Una señal modulada en PWM es una señal cuadrada de frecuencia constante, cuya relación entre el tiempo de la parte positiva (‘1’) y el tiempo total se determina con el ciclo de trabajo. De esa forma, un ciclo de trabajo unitario determina que la señal PWM está siempre a ‘1’, mientras que un ciclo de trabajo nulo determina que la señal está siempre a ‘0’.
- Regulador: utiliza el PWM generado para controlar la planta. En el caso de un regulador en lazo cerrado, éste calcula el error entre el valor esperado y el actual de la planta (tensión de salida) y obtiene el valor de la actuación a aplicar para eliminar el posible error. Ese nuevo valor de la actuación será el nuevo ciclo de trabajo que se aplicará a la señal PWM.
- Gestor ADC: se encarga de suministrar al regulador el valor actual de la señal a regular de la planta en instantes concretos de tiempo, cuando el dato obtenido por el ADC (*Analog to Digital Converter*) sea válido.

En el laboratorio de la asignatura, la consigna deseada para el sistema es introducida manualmente mediante los interruptores de la FPGA. Por otra parte, mediante sus botones es posible realizar un Reset, elegir modo de funcionamiento (lazo abierto o cerrado) o activar el proceso de regulación. Inicialmente los alumnos simulan conjuntamente el regulador y la planta digital en un lenguaje de descripción *hardware* (HDL) para detectar posibles errores antes de probarlos en la planta analógica, utilizando técnicas similares a las vistas en [6].

El resultado del laboratorio remoto para este sistema de control es visualizar la monitorización de la salida de tensión de la planta con un osciloscopio integrado en la Web y poder asignar remotamente los datos de control que antes se asignaban manualmente.

1.1. Motivación

Las sesiones prácticas en laboratorios tradicionales disponen de un tiempo limitado y muchas veces insuficiente para alcanzar todos los objetivos propuestos, necesitando dedicación extra-horaria de las sesiones. Otro factor limitante es el elevado precio de los dispositivos utilizados y sus licencias para los alumnos, haciendo complicada su adquisición para uso personal. Este problema de disponibilidad y difícil acceso resultan en un aprendizaje condicionado.

Con el desarrollo de este TFG se resuelven los problemas de disponibilidad y espacio-temporales restringidos por el laboratorio tradicional. El laboratorio remoto desarrollado permite al alumno programar el dispositivo digital programable (en este caso una FPGA) con sus ficheros fuente, asignar los valores (que anteriormente se introducían mediante una botonera) a través de una interfaz web y observar el funcionamiento del sistema en tiempo real, de manera remota y a cualquier hora del día.

Asimismo, se solventa el problema de montaje *hardware*, fuente potencial de errores. El alumno no tiene que preocuparse a la hora de diseñar el regulador de si el circuito, osciloscopio o FPGA están bien o mal interconectados, pues ya le son proporcionados en correcto funcionamiento. Aislando al alumno de estas fuentes de error y una vez verificado su regulador con la interfaz web, deberá realizar su propio montaje y comprobar de forma presencial un funcionamiento afín al del laboratorio remoto.

Se han presentado interesantes propuestas de laboratorios remotos para ingenierías [7 , 8 , 9] así como resultados satisfactorios obtenidos mediante encuestas posteriores al uso de laboratorios remotos por parte de los alumnos, donde destacan la flexibilidad que les proporciona y mejora en el aprendizaje [10 , 11 , 12].

1.2. Objetivos

Los principales objetivos de este TFG son:

- Desarrollo de una interfaz web accesible desde cualquier ordenador con conexión a Internet y un navegador con Java activado. Síntesis de los códigos fuente de los alumnos y programación de la FPGA, sin necesidad de que el alumno tenga un programa de síntesis en su ordenador personal.
- Interacción entre la FPGA y la interfaz web para poder cambiar la consigna de actuación, mostrar valores de depuración, etc. Esta interacción se realizará mediante un protocolo UART, y un convertidor USB-UART para que el servidor web no necesite tener puertos COM externos.
- Visualización de la salida del osciloscopio integrándolo en la interfaz web.
- Desarrollo de un sistema de reservas para los alumnos y limitación de tiempo de uso de la interfaz.
- Una correcta actualización y fácil gestión de la base de datos.

1.3. Estructuración de la memoria

El **primer capítulo** ha expuesto la introducción del TFG, motivaciones de su realización y consecuciones perseguidas.

En el **segundo capítulo** se expone un esquema general de los módulos que forman el proyecto, su disposición final y elementos que lo componen.

En el **tercer capítulo** se describen cada uno de los elementos y tecnologías que han sido utilizadas para el desarrollo.

En el **cuarto capítulo** se muestran los detalles de diseño, implementación y las pruebas y resultados de todo el conjunto del proyecto en funcionamiento.

En el **quinto capítulo** se exponen las conclusiones obtenidas y posibles líneas de desarrollo futuro.

2. ESQUEMA GENERAL

El sistema se ha basado en una arquitectura cliente-servidor como muestra la Figura 2. El servidor web se encuentra en un laboratorio de la Escuela Politécnica Superior y a él se conectan los siguientes elementos:

- Un osciloscopio mediante un cable Ethernet
- Un conversor USB-UART
- Una FPGA mediante un cable JTAG-USB

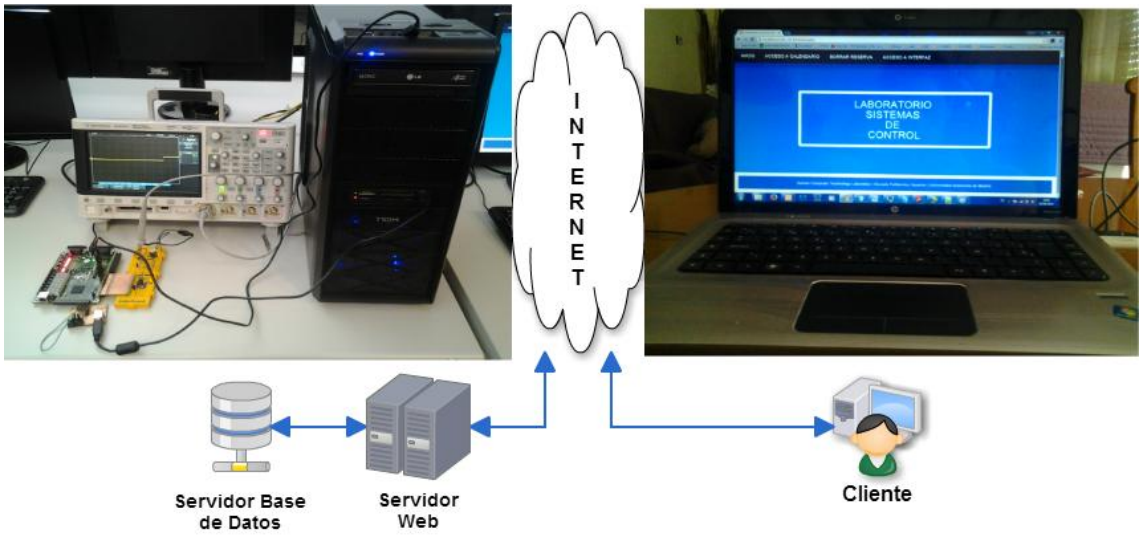

Figura 2. Arquitectura final del sistema implementado

Un conversor USB-UART y el circuito electrónico son conectados a la FPGA como se observa en la Figura 3 y una sonda pasiva medirá la tensión de salida del circuito electrónico para su visualización en el osciloscopio.

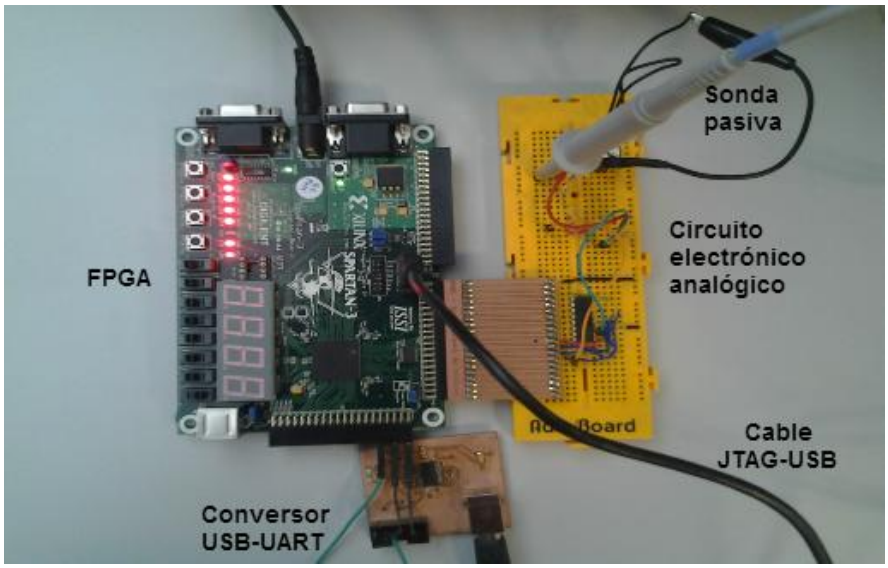

Figura 3. Módulos conectados a la FPGA

Como se explicó anteriormente, el circuito electrónico sencillo actuará como control de la tensión de salida y para este laboratorio en concreto, se podrá tener dos modelos de planta que se conectarán en la placa de conexiones como muestra la Figura 4:

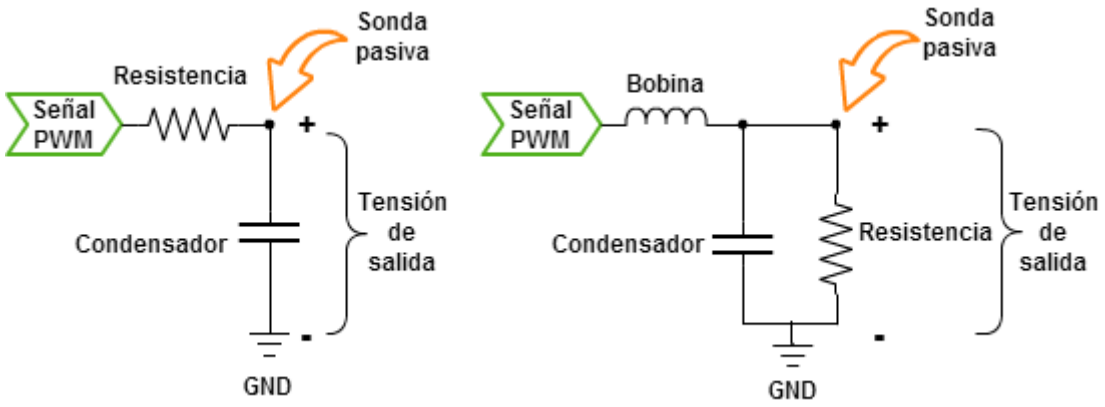

Figura 4. Modelo de planta RC (izq.) y RLC (dcha.)

Por último, mencionar que en el servidor web deberán alojarse los archivos que se detallan en la Figura 5 y pueden consultarse en el Anexo I.1. Estos archivos junto con los que envía el cliente al servidor, son necesarios para el funcionamiento del laboratorio remoto.

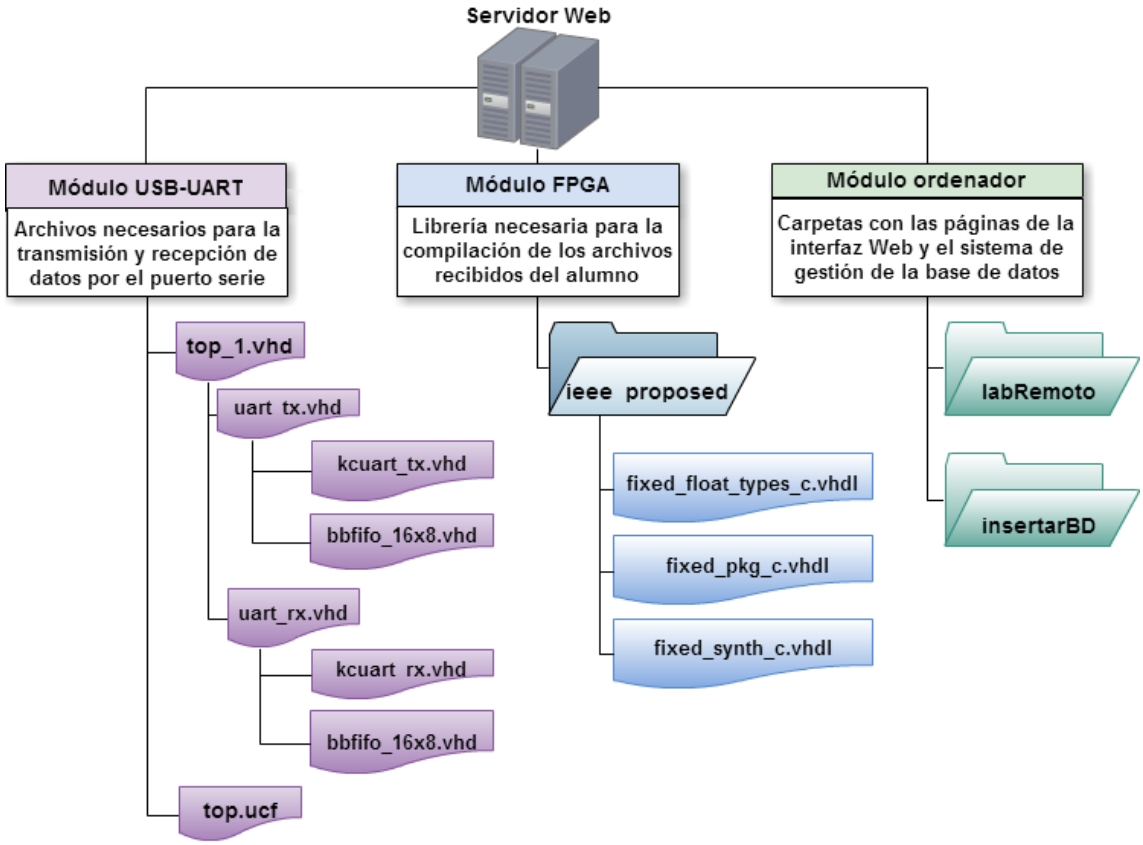

Figura 5. Esquema con los archivos que aloja el servidor

3. TECNOLOGÍAS Y DISPOSITIVOS USADOS

Una vez descrito el esquema general y puesto que se desarrolla en torno a una aplicación web, se ha decidido dividir en las tecnologías y dispositivos que se han escogido según se empleen en el servidor o en el cliente.

3.1. Tecnologías y dispositivos en la parte del servidor

Para desarrollar toda la funcionalidad ha sido necesario integrar con software y *hardware* especializado al servidor que actuará de base, valorando las ventajas que nos puede proporcionar a cada uno de los módulos. El Sistema Operativo escogido ha sido *Windows 7 Service Pack 1*. A continuación se detallan los elementos que han sido necesarios.

3.1.1 Desarrollo en FPGA

Es un dispositivo digital reprogramable cuyos bloques de lógica (*CLB: Configurable Logic Blocks*) están interconectados y pueden ser reconfigurados mediante un lenguaje HDL, como puede ser VHDL o Verilog entre otros. Ofrecen rápido prototipado, flexibilidad en los diseños, y un buen rendimiento por la capacidad de realizar múltiples operaciones paralelamente. En el desarrollo de este TFG se ha utilizado una *FPGA Xilinx Spartan 3* integrada en la placa de desarrollo *Spartan-3 Starter Board* de Digilent (ver Figura 6) compatible con todas las herramientas de Xilinx ISE y un cable de programación JTAG-USB para la conexión al ordenador y poder programar el fichero *bitstream* .bit.

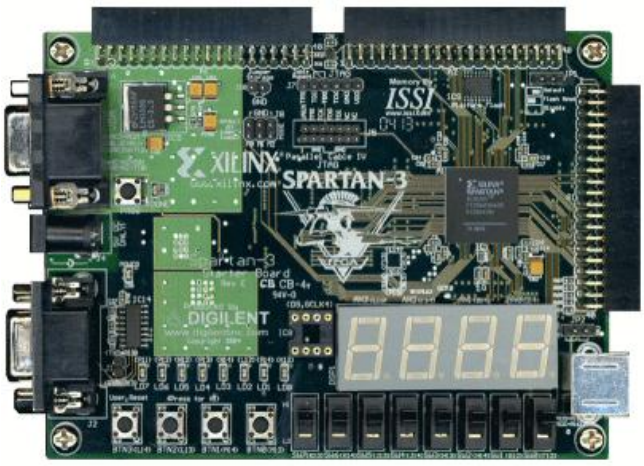

Figura 6. Placa de desarrollo Spartan-3 Starter Board

El circuito que contiene el circuito electrónico (circuito RC o RLC) y el ADC se conecta a uno de los bloques de pines de expansión y el módulo USB-UART es asignado a

otro, manteniendo las asignaciones de los pines que se describen en el UCF (*User Constraints File*), un fichero que mapea los pines de entrada y salida de la FPGA con el circuito implementado.

Una vez conectada con los módulos y programada con los ficheros que suministra el alumno desde la interfaz más los que se proporcionan para permitir la comunicación con el puerto serie, se encarga de:

- Mostrar mediante los 8 LEDs el valor del dato (8 bits) obtenido por el ADC.
- Recibir por el pin asignado como canal de entrada de la UART señales de Reset, selector de lazo abierto/cerrado y la consigna de regulación que el alumno selecciona desde la interfaz web. Estos datos son determinantes para el funcionamiento del controlador y planta. Anteriormente se asignaban de forma manual en el laboratorio.
- Enviar por el pin asignado de canal de salida de la UART el dato leído del ADC, es decir, lo mismo que se vería en los LEDs de la FPGA en el laboratorio tradicional. Este dato se envía periódicamente para ser mostrado y actualizado automáticamente en la interfaz web.

3.1.2 Desarrollo en servidor web

SERVIDOR WEB

i. XAMPP (Apache + MySQL + PHP+ Perl)

XAMMP [13] es un paquete de distribución multiplataforma y de software libre que contiene entre sus elementos tres de los cuales se utilizan para el desarrollo de la interfaz: servidor web, un Sistema de Gestión de Bases de Datos (SGBD) y el módulo para PHP *(PHP Hypertext Pre-processor).*

ii. Servidor web Apache

Un servidor web nos permite transferir datos de hipertexto (páginas web con todos sus elementos) utilizando el protocolo HTTP, archivos (mediante *File Transfer Protocol*, FTP) o correos entre otras cosas. En este caso nuestro servidor web se encontrará alojado en un ordenador conectado a la Escuela Politécnica Superior a la espera de que un cliente realice una petición de una página web, la cual traduce a código HTML y la transfiere mediante Internet respondiendo a la petición del cliente. Se ha escogido el servidor Apache [14] por su arquitectura modular, multiplataforma, opciones de seguridad, su continuo

desarrollo y una fácil configuración a través de directivas en un fichero que se detallarán en el Anexo I.1.

iii. PHP

PHP [15] es un lenguaje de programación diseñado para el desarrollo web de contenido dinámico que se ejecuta en el lado del servidor. Este lenguaje de alto nivel ha sido escogido para desarrollar la interfaz web por la facilidad con que puede incrustarse en código HTML, el procesado de formularios y fácil generación de ficheros, el soporte para distintas bases de datos y la posibilidad de programación por procedimientos u orientado a objetos (POO). El procedimiento con el que actúa PHP es que tras recibir una solicitud del cliente, el servidor procesa la petición y prepara la información necesaria (consultando una base de datos, por ejemplo) y envía al cliente una página web estática, pero cuya creación ha sido dinámica (por los procesos que realiza el servidor, de modo que la respuesta no es siempre la misma).

i. NetBeans IDE 8.0.2

NetBeans [16] es un entorno de desarrollo integrado libre (IDE) y multiplataforma. Permite desarrollar aplicaciones, compilar y depurar código según los módulos que se instalen. Para este TFG se ha hecho uso de los módulos HTML5 y PHP.

ii. Microsoft Visual C++ 2012 y lenguaje C++

Como se ha comentado anteriormente, existe un canal bidireccional de comunicación basado en UART entre el servidor web y la FPGA que actúa de regulador del circuito. En el caso de escritura, PHP en Windows puede mandar información sin ningún tipo de dificultad técnica. Sin embargo, en la lectura, no hay soporte directo para PHP. Por tanto, se ha realizado una aplicación en lenguaje C++. Se ha hecho uso del IDE Visual C++ creando un programa cuya misión es recibir un byte por el puerto serie y devuelve su valor como retorno del programa para mostrarlo en la interfaz web. La ejecución de una nueva instancia de un programa, en vez de utilizar otros métodos como sockets, requiere un gasto computacional no despreciable. Aún así, el diseño elegido es totalmente viable ya que este programa se ejecutará una vez cada 5 segundos (valor que puede modificarse en el código fuente, ver Anexo II.2.14) cabeceraLab_1.inc y II.2.15) lecturaPuerto.php) y su ejecución dura escasos milisegundos.

BASE DE DATOS

i. MySQL

Es un SGBD que utiliza declaraciones basadas en el Lenguaje de Consulta Estructurado (*SQL* en inglés). En este TFG ha sido necesario implementar una base de datos con dos tablas: una para mantener los datos de los alumnos y otro para el sistema de reservas de acceso a la interfaz web, restringiendo el acceso a un alumno por horario de reserva. El motor de almacenamiento escogido ha sido *InnoDB* [17], pues ofrece más fiabilidad, mejor rendimiento, recuperación de fallos y realiza las transacciones atendiendo a los principios de atomicidad, consistencia, aislamiento y durabilidad (*ACID* en inglés) frente a otros motores.

Para la administración de MySQL se ha utilizado la interfaz gráfica **phpMyAdmin** [18] incluida en el paquete XAMPP (ver Figura 7). Permite crear de una forma sencilla usuarios, asignar permisos, gestionar las bases de datos, tablas, exportarlas, establecer relaciones entre ellas, realizar copias de seguridad, etc.

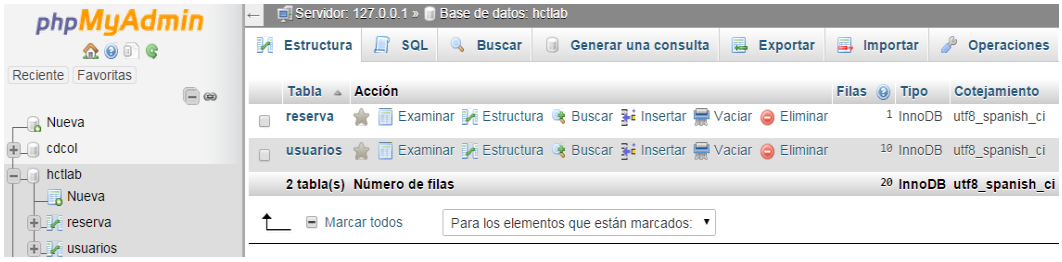

Figura 7. Interfaz gráfica de phpMyAdmin

HERRAMIENTAS HDL

i. ModelSim PE Student Edition 4a

ModelSim [18] es un simulador de múltiples Lenguajes de Descripción *Hardware* (*HDL* en inglés) como VHDL o Verilog. Permite simulaciones a nivel de puertas lógicas para diseños de circuitos digitales en FPGAs, entre otros dispositivos, teniendo en cuenta nociones de tiempo. Se utilizan librerías (directorios con unidades de diseño compiladas) que compilan los circuitos descritos y el resultado se puede mostrar en una interfaz gráfica al usuario, además de poder realizar simulaciones en batería con fines de verificación automatizada.

En este TFG ModelSim se ha utilizado como analizador de sintaxis y detector de errores básicos de compilación de los ficheros descritos en VHDL que el alumno sube a la interfaz web. Se le permite escoger o no una librería necesaria para algunos diseños y

utilizando PHP ejecuta la línea de comandos del simulador. En caso de existir errores los muestra en la web al alumno.

ii. Xilinx ISE

Xilinx ISE [19] es una herramienta software bajo licencia que asiste al proceso de diseño, simulación, síntesis del resultado, implementación y configuración de dispositivos reprogramables de Xilinx, entre las que se encuentran las FPGAs. Para nuestra funcionalidad, se ha utilizado Xilinx ISE para realizar el proceso de síntesis e implementación de la lógica de diseños VHDL junto con las librerías adecuadas a un archivo de programación (*bitstream* o .bit). Ese archivo .bit permite la programación de la FPGA con la invocación del programa *ISE iMPACT*. Para llegar a este punto desde la interfaz web, Xilinx provee el programa *XFLOW***,** una herramienta de línea de comandos que permite automatizar todo lo anterior y que se describirá en el siguiente capítulo.

3.1.3 Módulo del circuito electrónico analógico

Se compone de una placa protoboard con el modelo de planta seleccionado (RC o RLC), un ADC del fabricante *Analog Devices* AD7813Y y una placa para conexiones de 20 pines especial para conectar a un bloque de pines de expansión de la FPGA.

3.1.4 Módulo de comunicación UART

Una UART (*Universal Asynchronous Receiver-Transmitter*) es un circuito integrado para puertos serie capaz de recibir y transmitir datos de forma serial realizando conversiones de bytes a secuencias de bits y viceversa. La comunicación más básica que puede implementar entre dos dispositivos ambos con una referencia común a tierra (GND), es que el pin transmisor (TX) del dispositivo 1 esté conectado al pin receptor (RX) del 2 y que el pin transmisor de este último esté conectado al receptor del dispositivo 1 (ver Figura 8).

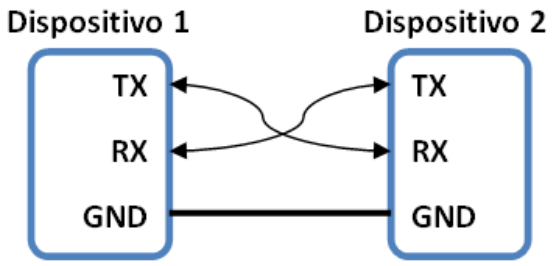

Figura 8. Esquema del bus de comunicación UART

La comunicación asíncrona significa que los dispositivos no están sincronizados por una señal de reloj común, pero ambos deben utilizar una referencia temporal con cierta tolerancia para permitir la transferencia serial de cada byte de datos.

La descripción de *Universal* significa que puede ser configurada para soportar diferentes protocolos específicos para puertos serie. Para la implementación en la FPGA, se ha reutilizado el módulo UART del microcontrolador softcore PicoBlaze de 8 bits [21], que provee Xilinx y puede ser implementado en la FPGA. Su funcionamiento y diseño se explica en el siguiente capítulo.

3.1.5 Módulo del osciloscopio

Se ha utilizado el osciloscopio *InfiniiVision MSO-X 3014A* [22] de la marca *Keysight Technologies* (ver Figura 9). Se ha configurado el montaje para capturar la tensión de salida de la planta mediante una sonda pasiva en el circuito. La elección de este osciloscopio ha sido determinada por dos funcionalidades, incluidas bajo licencia, necesarias para este TFG:

- Disparo y decodificación de bus serial UART/RS232: ha permitido verificar el correcto envío y recepción de tramas a través del puerto serie configurado. Esta característica sólo ha sido necesaria durante el diseño de este TFG para poder depurar el envío de tramas, no siendo necesaria esta característica para su posterior explotación.
- Módulo DSOXLAN LAN/VGA (ver Figura 10): permite hacer uso de la interfaz web que ofrece el osciloscopio con una conexión Ethernet.

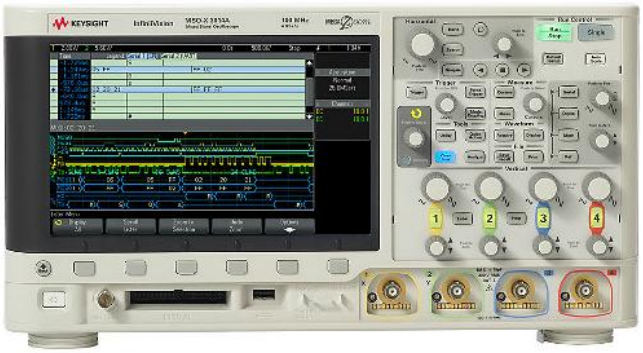

Figura 9. Osciloscopio MSO-X 3014A

Figura 10.Parte posterior con módulo LAN/VGA

3.2. Tecnologías y dispositivos en la parte del cliente

Lo componen las líneas de código que debe interpretar el ordenador de los alumnos que accedan a la interfaz web. El servidor envía el código HTML que puede incluir código en lenguaje de script, y el navegador es el encargado de interpretar y dar formato a la página. La principal ventaja de que se ejecute en el cliente es una menor carga de recursos al servidor, además permitir implementar contenidos dinámicos sin necesidad de recargar toda la página web.

3.2.1 CSS

CSS (*Cascading Style Sheets*) es un lenguaje de hojas de estilo para definir y controlar el aspecto y presentación de las páginas web en los navegadores. Permite separar los contenidos de la web (párrafos, titulares, tablas, listas de elementos, etc) del estilo que se quiere aplicar a cada uno, modificando sólo ciertos parámetros de CSS (tamaño del texto, color, posición de cada elemento en la página, etc).

Para el diseño de la interfaz se ha escogido un *framework* llamado Pure CSS, que permite su uso por módulos diferenciados, su código es muy ligero y funciona de manera responsiva, adaptándose a los distintos tamaños de pantalla.

3.2.2 JavaScript

Lenguaje de programación para la creación de webs dinámicas que interaccionen con el usuario. No necesita ser compilado, sino que cada línea de código se traduce a lenguaje máquina antes de ejecutarse. En la interfaz web se ha utilizado un script para controlar el tiempo en la interfaz web y cerrar sesión cuando se alcance el límite de permanencia.

3.2.3 jQuery y AJAX

jQuery es una librería con funcionalidades de JavaScript que simplifica la manera de interactuar con los documentos HTML, animaciones, gestión de eventos e interacciones AJAX. AJAX es utilizado para intercambiar datos con el servidor y actualizar partes asíncronamente de una página web sin tener que recargar la página entera de nuevo. En la Figura 11 se muestra un esquema comparando una petición clásica y otra con AJAX. Con la combinación de ambas tecnologías se ha conseguido mostrar datos de la lectura del puerto serie de forma periódica y automática.

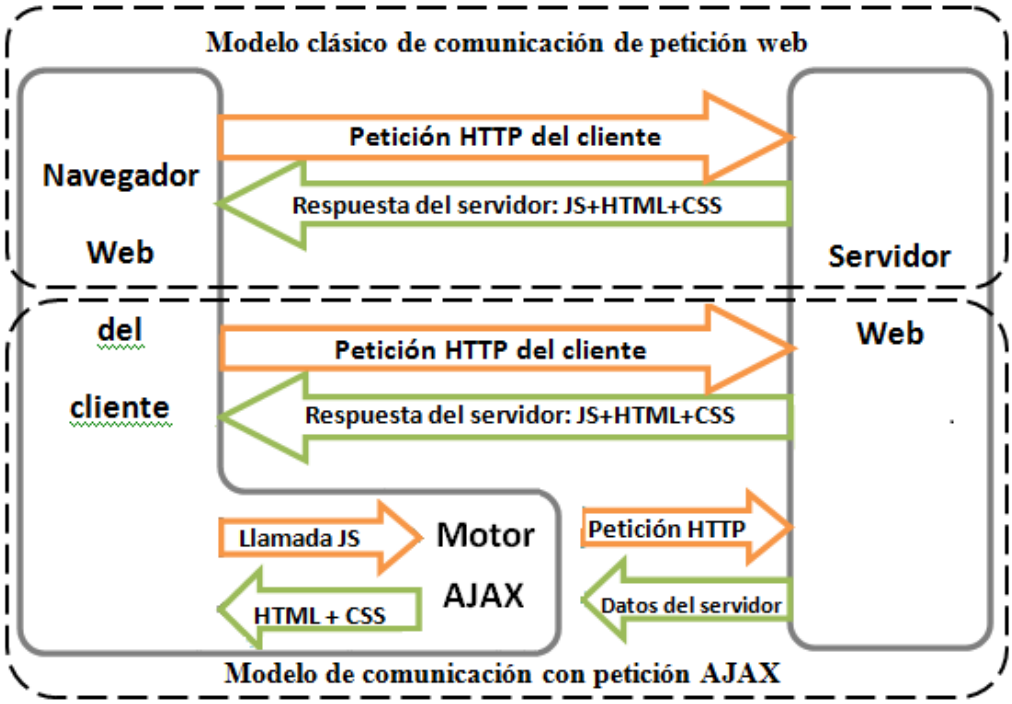

Figura 11 Comparación entre petición web clásica y AJAX

Para poder usar estas 3 tecnologías se han incluido en las líneas de cabecera de cada página web la CDN (*Content Delivery Network*) o Red de Distribución de Contenido más actual de cada uno (Figura 12). Una CDN es un conjunto de servidores ubicados en diferentes puntos de la red y que contienen copias locales de ciertos contenidos. Llamando a las CDNs que necesitamos para hacer uso de estas tecnologías, evitamos sobrecargar el servidor, mejorar la velocidad de acceso y reducir la latencia.

También se ha hecho uso de *Font Awesome* una tipografía que incluye iconos especiales para la web y el archivo `style.css`, desarrollado para dar formato a todas las páginas.

```
<link rel="stylesheet" href="http://netdna.bootstrapcdn.com/font-awesome/4.0.3/css/font-awesome.css">
<link rel="stylesheet" href="http://yui.yahooapis.com/pure/0.6.0/grids-responsive-min.css">
<link rel="stylesheet" href="http://yui.yahooapis.com/pure/0.6.0/pure-min.css">
<script src="https://code.jquery.com/jquery-1.11.3.min.js"></script>
<script src="//code.jquery.com/ui/1.11.4/jquery-ui.js"></script>
<link rel="stylesheet" href="//code.jquery.com/ui/1.11.4/themes/smoothness/jquery-ui.css">
<link rel="stylesheet" href="/Lab_10_02/include/style.css">
```

Figura 12. CDN de las tecnologías utilizadas y fichero CSS general de la interfaz web

4. Diseño, implementación y pruebas

Una vez explicados todos los módulos y tecnologías necesarias se procede a mostrar el diseño, implementación y pruebas realizadas para el desarrollo de la interfaz. Primero se explicará el diseño del módulo UART para la comunicación con la FPGA, posteriormente la implementación de la base de datos y por último se mostrará el funcionamiento conjunto con la interfaz web desarrollada.

4.1 UART

El modo de funcionamiento asíncrono consiste en que los datos son transmitidos de forma serial empezando por el bit menos significativo (*LSB*). Puesto que el transmisor puede empezar a enviar datos en cualquier momento, el receptor necesita un método de identificar cuándo ese *LSB* es enviado. Para ello, el transmisor envía un bit de *start* a nivel activo bajo como muestra la Figura 13. El receptor utiliza ese flanco de bajada para muestrear los valores de los bits entrantes en un punto aproximado a la mitad de la duración del pulso de cada bit recibido, donde el dato es más estable (ver Figura 14). Después que el bit más significativo (*MSB*) ha sido muestreado, el receptor espera que llegue el bit de *stop* activo alto para confirmar el fin de la transmisión. Cuando la línea está a la espera de datos también se mantiene activo alto.

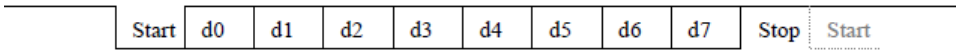

Figura 13. Transmisión serial de los bits

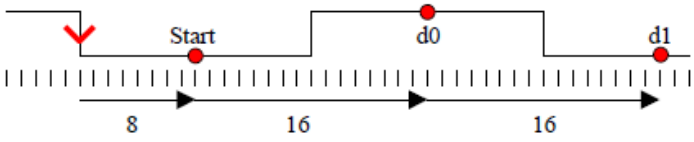

Figura 14. Ejemplo de muestreo de los datos entrantes en el receptor

Los parámetros que se han configurado en ambos dispositivos para que la comunicación asíncrona tenga éxito son:

- **Velocidad de transmisión en baudios (*baud rate*):** los módulos utilizados funcionan desde 9600, 19200, 38400, 57600 baudios en adelante hasta 10 Mbaudios o Mbit por segundo. Se ha escogido una velocidad estándar de 19200 baudios.
- **Número de bits del dato:** pueden formarse longitudes de 5, 6, 7, 8 e incluso 9 bits. En este caso, escogemos el dato estándar de 8 bits.
- **Bits de *stop*:** indican el fin de transmisión del dato enviado. Para este TFG se ha configurado un único bit de stop, aunque también soporta 2 bits de stop.

- **Paridad:** es un sencillo método de detección de errores. Puede escogerse entre paridad par, impar, fija o no usar. Para esta UART no se ha utilizado paridad.

Los módulos de UART utilizados esperan que la referencia temporal sea una señal 16 veces más rápida (llamada *en_16_x_baud*) que la velocidad de transmisión en baudios. Esta señal debe durar únicamente un ciclo de reloj del reloj principal del módulo UART, que en el diseño propuesto tiene una frecuencia de 50 Mhz (ver Figura 15).

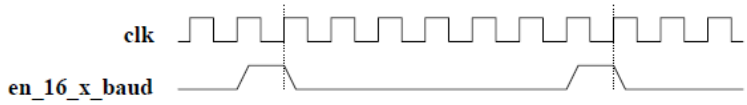

Figura 15. Sincronización entre reloj y señal de habilitación

El reloj físico de la FPGA funciona a 50MHz, el *baud rate* deseado son 19200 baudios (o bits por segundo) y la señal *en_16_ex_baud* debe conmutar 16 veces más rápido:

$$\frac{50\ MHz}{16 \times 19200} = 162{,}76 \sim 163$$

Esto significa que cada 163 ciclos de reloj de la FPGA se debe activar durante un pulso la señal *en_16_ex_baud*. Con esa aproximación, la UART estaría funcionando a $19171,78$ baudios, lo que conlleva un error de:

$$\left|\frac{19171{,}78 - 19200}{19200}\right| = 0{,}00147 = 0{,}147\%$$

Se puede considerar que un error por debajo del 1% no influirá negativamente en el diseño, por lo que la velocidad escogida es adecuada. Además, hay que tener en cuenta que en nuestro diseño, nunca se enviarán dos bytes seguidos sin haber un tiempo de guarda entre ellos. Por tanto, este error se reiniciaría entre cada transmisión. En VHDL esta especificación de la señal se ha realizado con el siguiente código de la Figura 16.

```
Baud_divisor : process(Clk, Reset)
begin
    if Reset = '1' then
        baud_count <= 0;
        enable_baud <= '0';
    elsif (clk'event and Clk='1') then
        if baud_count = 162 then
          baud_count <= 0;
          enable_baud <= '1';
      else
          baud_count <= baud_count + 1;
          enable_baud <= '0';
        end if;
   end if;
end process Baud_divisor;
```

Figura 16. Proceso para obtener la referencia temporal 'en_16_ex_baud'

4.1.1 Transmisión UART

El bloque para la transmisión (UART_TX) se compone de 3 archivos en el proyecto de Xilinx: el *top level* lo implementa *uart_tx* englobando al transmisor compacto constante *kcuart_tx.vhd* y un buffer FIFO para las tramas denominado *bbfifo_16x8.vhd* (ver Figura 17 y Figura 18).

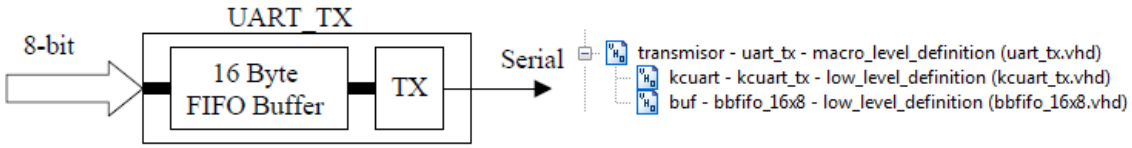

Figura 17. Buffer FIFO embebido en el transmisor (izq.) y jerarquía de archivos (dcha.)

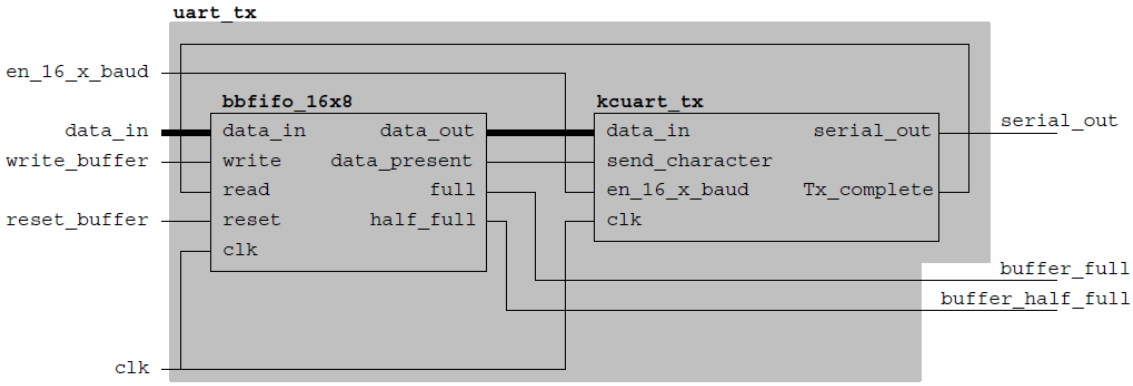

Figura 18. Esquema de señales del top level uart_tx.vhd

A continuación se describen las más importantes que se han utilizado en este TFG para la transmisión de FPGA a PC.

`data_in`: son los 8 bits de entrada para transmitir en serie. Los datos son capturados por el buffer *FIFO* en flanco de subida de reloj y cuando `write_buffer = '1'`.

`write_buffer:` si está activo a '1', significa que los datos se están aplicando a `data_in` serán escritos en el buffer interno en el próximo flanco de subida de reloj. Cada operación de escritura tomará un ciclo de reloj, por lo que debería ser activada un único ciclo a no ser que nuevos datos estén llegando a `data_in,` para evitar el llenado del buffer. Si el buffer ha excedido su capacidad, ignorará los datos que sigan llegando.

`en_16_x_baud:` es la señal con frecuencia 16 veces mayor que la velocidad de transmisión (*baud rate*) que provee la referencia temporal para la transmisión serial.

`serial_out:` señal de salida del protocolo UART.

buffer_full y buffer_half_full: se activan cuando el buffer está lleno y a mitad de capacidad respectivamente. No han sido implementadas para este proyecto, pues al no enviar ráfagas de datos, sino bytes separados por tiempos de espera (ver Figura 19), nos aseguramos de que no se sature el buffer.

```
constant MAX_TX_COUNT : integer := 1000000; -- Cada 0.02s
-- Señal para controlar cada cuánto tiempo se envía un dato hacia la
interfaz web
signal write_count  : integer range 0 to MAX_TX_COUNT; -- (f = 50MHz
-> T = 20ns)
Buffer_contador : process(Clk, Reset)
begin
    if Reset = '1' then
            write_count  <= 0;
            write_buffer_tx <= '0';
    elsif(clk'event and Clk='1') then

        if(write_count < MAX_TX_COUNT) then
            write_count  <= write_count + 1;
            write_buffer_tx <= '0';
        else
            write_buffer_tx <= '1';
            write_count <= 0;
        end if;
    end if;
end process Buffer_contador;
```

Figura 19. Proceso VHDL que controla el tiempo entre dos envíos UART consecutivos

El dato que se desea transmitir es el mismo valor que el alumno vería en los LEDs si estuviera realizando la práctica de forma presencial. En particular, es útil que este dato sea el valor que registra el ADC. Como se ha explicado, este dato será enviado regularmente tanto por la UART cada vez que se active **write_buffer** según el proceso anterior como mostrado por la interfaz web.

4.1.2 Recepción UART

El bloque para la recepción (UART_RX) también se compone de tres archivos: el *top level* lo implementa *uart_rx* englobando al receptor compacto constante *kcuart_rx.vhd* y el buffer FIFO *bbfifo_16x8.vhd* como se observa en las Figura 20 y Figura 21.

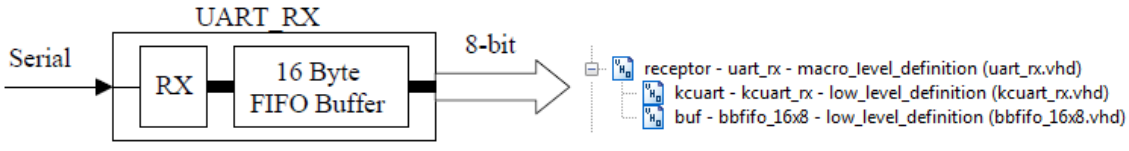

Figura 20. Buffer FIFO embebido en el receptor (izq.) y jerarquía de archivos (dcha.)

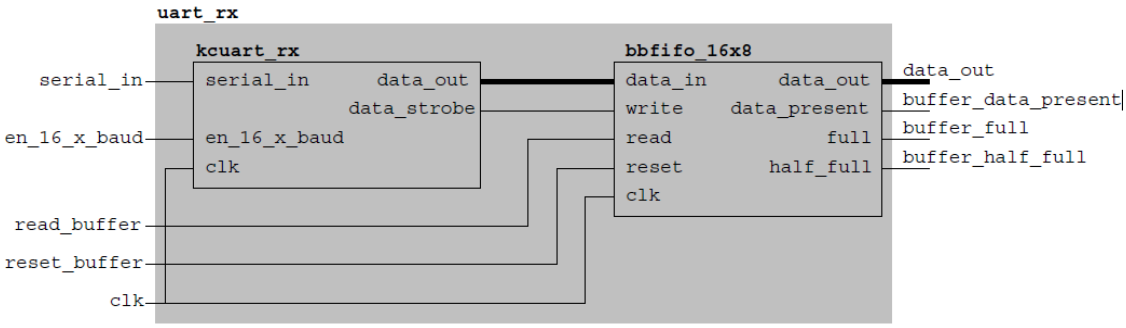

Figura 21. Esquema de señales del top level uart_rx.vhd

Como anteriormente, se describen las señales más importantes utilizadas en este TFG para la recepción de PC a FPGA.

`serial_in`: señal de entrada conformada con el protocolo UART configurado. En modo de espera de datos está a '1' y con el flanco de bajada, detecta el bit de *start* para el comienzo de la transmisión serial.

`data_out`: son los 8 bits de datos que se han recibido. El dato es válido cuando `buffer_data_present = '1'`.

`read_buffer`: cuando se activa a '1' significa que los datos que contiene `data_out` han sido leídos (o lo serán en el próximo flanco de subida de reloj), indicando que se está a la espera de recibir un nuevo byte.

`Buffer_data_present`: se activa a '1' automáticamente cuando el buffer *FIFO* contiene uno o más bytes de datos recibidos y los datos válidos estarán disponibles en `data_out`.

`buffer_full y buffer_half_full`: al igual que en el transmisor, estas señales no han sido utilizadas dado que se controlan los bytes recibidos como se explica a continuación.

El proceso de recepción de los datos enviados desde la web hacia la UART para que se asignen en la FPGA se ha compuesto de tres procesos.

El primero de ellos (Figura 22) lleva una cuenta del número de bytes que se han recibido. En cada ciclo de reloj, siempre que haya un dato presente en el buffer (señalizado por `buffer_data_present = '1'`) incrementamos el contador hasta 3, que es el número de bytes que esperamos recibir desde la web.

```
numeroBytesRecibidos : process(Clk,Reset)
begin
    if Reset = '1' then
        uart_rx_bytes_received <= 0;

    elsif (clk'event and Clk='1') then
        if contadorTimeout = MAX_TIMEOUT then
            uart_rx_bytes_received <= 0;
        elsif uart_rx_bytes_received = 3 then
            uart_rx_bytes_received <= 0;
```

```
        else
            if buffer_data_present_rx = '1' then
                uart_rx_bytes_received <= uart_rx_bytes_received + 1;
            end if;
        end if;
    end if;
end process numeroBytesRecibidos;
```

Figura 22. Proceso VHDL para el número de bytes recibidos

Para enviar la trama completa de los 3 bytes se ha creado un *array* de 3 elementos y cada uno de ellos de 8 bits de longitud como muestra la Figura 23.

```
type RX_ARRAY is array(2 downto 0) of std_logic_vector(7 downto 0);
signal web_rx_frame: RX_ARRAY;
```

Figura 23. Señal definida para almacenar los datos recibidos por 'data_out'

En el siguiente proceso (Figura 24) cuando hay un dato recibido en el buffer, la salida `data_out` se asigna a cada una de las posiciones del *array* creado. El orden y correcta asignación se mantiene gracias al proceso explicado anteriormente, así como se desea leer varios datos, la señal `read_buffer` se mantiene activa a '1' (ver Figura 25). Cuando el *array* se ha completado con los 3 bytes, la señal `trama_completa = '1'` afirma que los 3 bytes se han recibido y están preparados para asignarse a sus señales correspondientes.

```
recepcionDatos: process(Clk, Reset)
    begin
        if Reset = '1' then
            trama_completa <= '0';
            for i in 0 to 2 loop
                web_rx_frame(i) <= (others => '0');
            end loop;
        elsif (clk'event and Clk='1') then
            -- Si hay dato recibido en el buffer
            if buffer_data_present_rx = '1' then
                -- Lo asignamos a una posición del array de datos
                web_rx_frame(uart_rx_bytes_received) <= data_out_rx;
                read_buffer_rx <= '1';
                    -- Hemos recibido todos los bytes
                if uart_rx_bytes_received = 2 then
                    -- Enviamos la trama
                    trama_completa <= '1';
                else
                    trama_completa <= '0';
                end if;
            else
                trama_completa <= '0';
            end if;
        end if;
end process recepcionDatos;
```

Figura 24. Proceso VHDL de asignación de los bytes recibidos

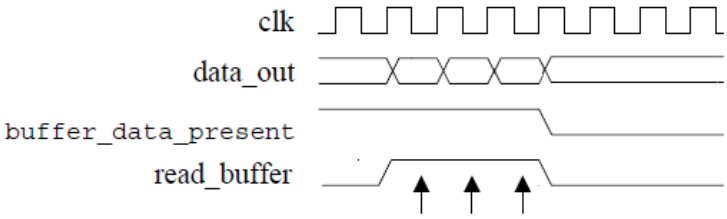

Figura 25. Esquema de activación de señales para lectura consecutiva de datos

El último proceso (Figura 26) se encarga de asignar a las señales del controlador los valores recibidos desde la interfaz web tras haberse recibido los 3 bytes indicado por `trama_completa = '1'`. Siempre se asignan en el mismo orden y en el caso de la señal *Swap* (que indica si el regulador debe funcionar como lazo abierto o cerrado), al ser solo de un bit, se le debe asignar el bit menos significativo que es el que en la web se ha configurado con el valor '0' ó '1' en el código PHP.

```
process(Clk,Reset)
begin
    if Reset = '1' then
        Swap <= '0';
        Consigna <= (others => '0');
        ResetUART <= '1';
    elsif (clk'event and Clk='1') then
        if trama_completa = '1' then
            Swap        <= web_rx_frame(0)(0);
            Consigna <= web_rx_frame(1);
            if web_rx_frame(2) = x"01" then
                ResetUART <= '1';
            else
                ResetUART <= '0';
            end if;
        else
            ResetUART <= '0';
        end if;
    end if;
end process;
```

Figura 26. Proceso VHDL de asignación de valores al controlador

La señal de `ResetUART` se asigna comparando el valor que se recibe en el último byte de la web. Esta señal se asigna de forma concurrente con una función lógica OR a otra señal (`Reset_regulador`) que está mapeada al `Reset` del controlador, pues no tendría sentido querer resetear la comunicación con la UART. La señal `Reset` del sistema completo sólo puede ser accedida manualmente desde la FPGA.

Por último, puesto que la UART espera recibir 3 bytes consecutivos, se ha creado un proceso con un contador a modo de *timeout.* Si en el tiempo de 1ms (MAX_TIMEOUT) no se ha recibido un nuevo byte para la trama actual que se está completando, reinicia el contador de bytes transmitidos, descartando así posibles tramas incompletas o ajenas a nuestro propósito. En la siguiente Figura 27 se muestra un ejemplo ilustrativo.

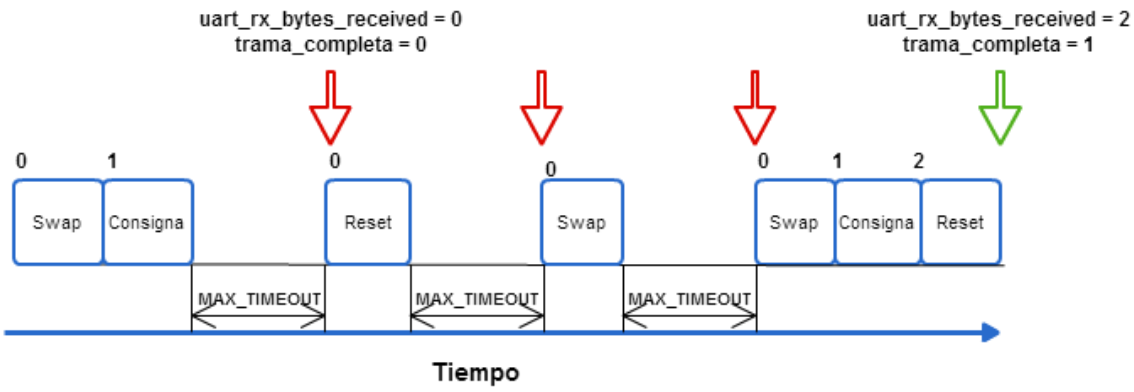

Figura 27. Esquema de timeout de bytes

4.1.3 Conexión USB-UART

La comunicación UART entre la FPGA y el servidor web se ha realizado mediante un circuito impreso que incluye un chip conversor USB-UART. El chip en concreto es un FT232 del fabricante FTDI (*Future Technology Devices International*). Las principales características que necesitábamos para este TFG son:

- Protocolo USB completamente manejado en el chip, sin necesidad de firmware adicional.
- Interfaz UART soportada para datos de 8 bits, 1 de *stop*.
- Compatible con USB 2.0 *full speed*.

Se han utilizado 3 pines de entrada/salida del conector de expansión B1 de la FPGA para conectar la UART (GND, TX y RX) entre la FPGA y el conversor mientras que para la conexión del PCB con el ordenador se ha utilizado un conector USB de tipo B.

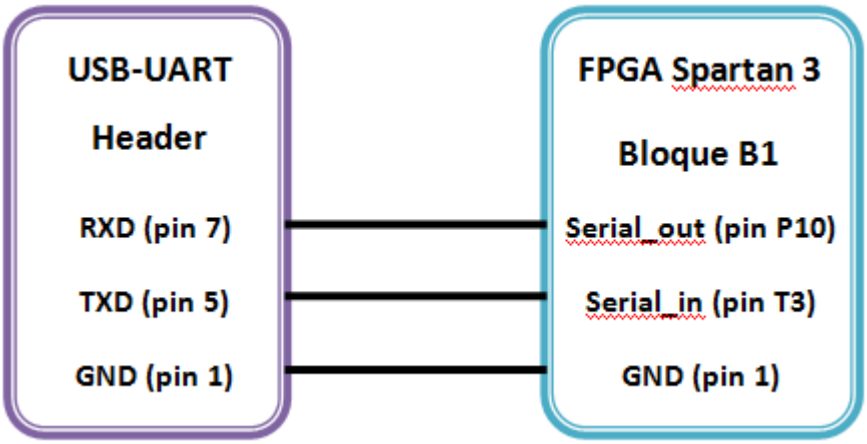

Figura 28. Conexiones entre el conector USB-UART y bloque de expansión B1 de la FPGA

Como se observa en la Figura 28, es importante que la señal de recepción RXD del USB-UART esté conectada al pin de salida de las tramas de la FPGA que deben ir hacia el ordenador, es decir, `serial_out`. Así como la señal de transmisión TXD debe conectarse a la entrada de recepción de la FPGA, `serial_in` por donde recibirá las tramas que se envían desde la interfaz web.

4.2 Base de datos

Como se ha comentado anteriormente, el servidor web debe ser compatible con un sistema de reservas para que los alumnos puedan reservar la hora a la que quieren acceder al sistema. Para ello se ha utilizado una base de datos MySQL que almacena dos tablas: una donde se guarda la información de los alumnos y otra con las reservas de dichos alumnos como se muestra en la Figura 29.

Figura 29. Estructura de las tablas de la base de datos

La tabla **usuarios** se compone de:

- id_usuario: es la clave primaria de la tabla. Identifica de forma única a cada registro (fila) de dicha tabla.
- email: campo para el correo electrónico del alumno. Estos datos los debe proveer el administrador de las prácticas.
- hash: campo que almacena el hash de la contraseña que se desea asignar al alumno. En el siguiente apartado se explicará cómo se almacena este dato.

La tabla **reserva** se compone de:

- id_reserva: clave primaria de la tabla con el mismo propósito que en la anterior.
- id_usuario: clave foránea que se relaciona con el campo de id_usuario de la tabla usuarios.
- email: campo para el correo electrónico del alumno que ha realizado la reserva de una hora para uso de la interfaz web.

- fecha_reserva: campo que mantiene la referencia a la fecha en que se realiza la reserva. Se proporciona desde la parte del servidor. La almacena con el tipo 'date', formato de fecha que incluye año, mes y día.
- acceso: indica si el alumno ha accedido o no a la interfaz con esa hora de reserva.
- hora_reserva: campo con formato 'time' para la hora en que se quiere acceder.

La base se ha diseñado de manera que sólo se permiten reservas para utilizarse en el mismo día que se realizan. También se hace un control de manera que un usuario no podrá realizar reservas consecutivas, sino que deberán transcurrir al menos 2 horas entre una y otra. Esto permite un control de manera que un usuario no monopolice la interfaz web. Estas especificaciones se realizan con el código PHP así como el tiempo de uso de la interfaz, dispuesto a 20 minutos. Los parámetros son designados como variables de configuración y pueden ser modificadas por el administrador a su elección (ver Anexo I.3).

4.3 Gestión de la Base de Datos

Para una fácil inserción de los alumnos que podrán acceder a la interfaz, se ha creado un módulo de gestión. Consiste en una web que solo podrá ser accedida desde el ordenador que actúe de servidor, manejada por el administrador de las prácticas. La Figura 30 muestra la pantalla de inicio y en la Figura 31 y Figura 32 las páginas que componen este módulo.

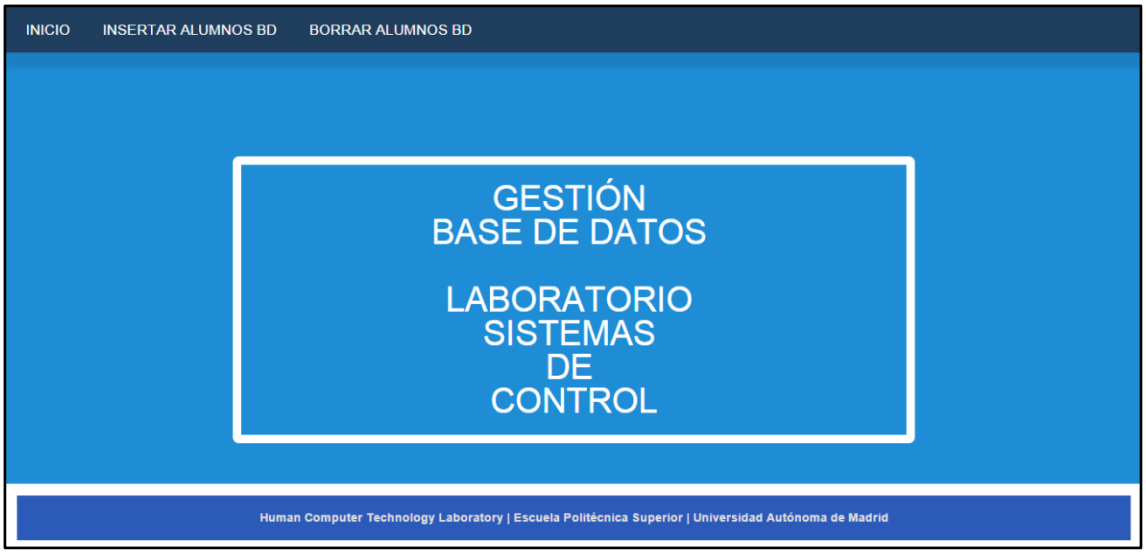

Figura 30. Captura de la pantalla de inicio del gestor para la base de datos

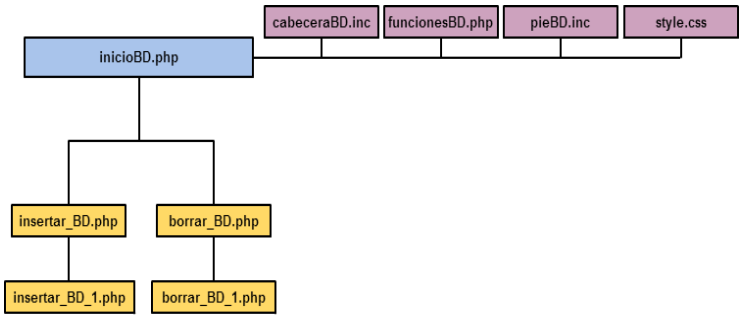

Figura 31. Esquema de la estructura para la Gestión de Base de Datos

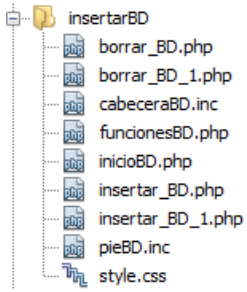

Figura 32. Directorio insertarBD

- **insertar_BD.php:** permite subir un fichero Excel con extensión .csv (*Comma Separated Values*) como muestra la captura de la Figura 33. Este formato permite obtener los datos con una coma como separador entre ellos, de manera que con PHP puede ser leído fácilmente y gestionado para introducir los datos en la base. El formato de este fichero debe tener dos columnas: la primera contendrá el correo institucional del alumno y la segunda la contraseña que se le asigna. La Figura 34 muestra un ejemplo.

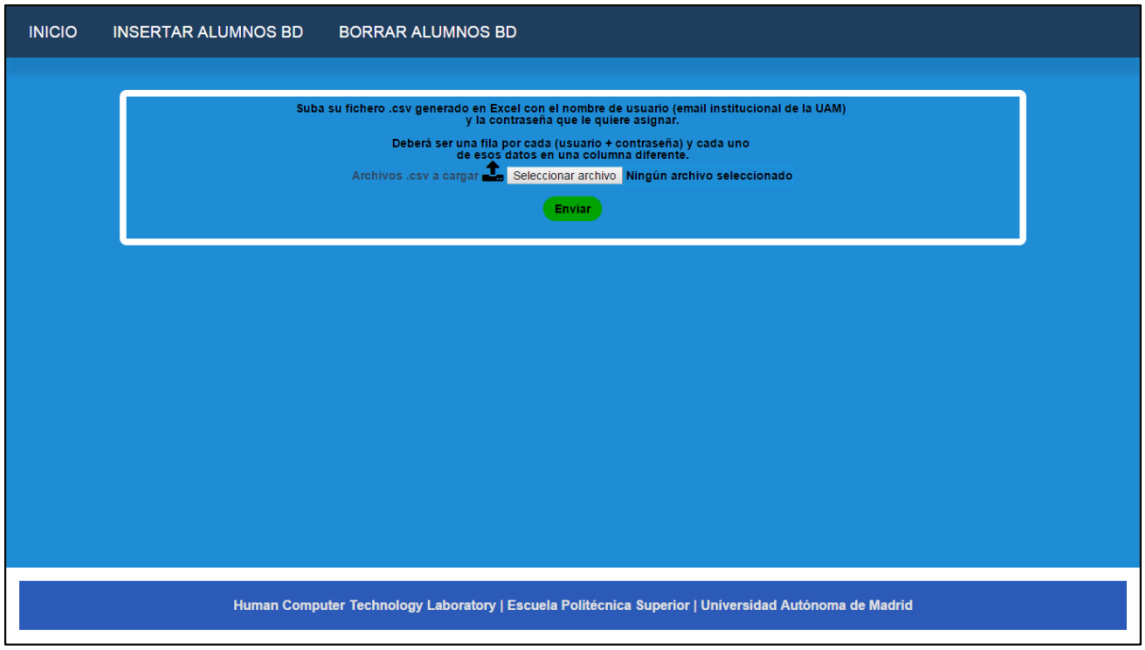

Figura 33. Captura de 'insertar_BD' para subir fichero .csv

	A	B
1	maria.murillom@estudiante.uam.es	1234
2	laura.murillom@estudiante.uam.es	5678
3	alexis.murillom@estudiante.uam.es	1290
4	dario.murillom@estudiante.uam.es	2345

Figura 34. Ejemplo de formato de datos para el fichero .csv

- **insertar_BD_1.php:** controla y gestiona el fichero subido. Si se sube un fichero no válido (con otra extensión que no sea .csv), si ya hay alumnos en la base de datos u ocurre algún error en la gestión se informa de ello y permite regresar al inicio (Figura 35).

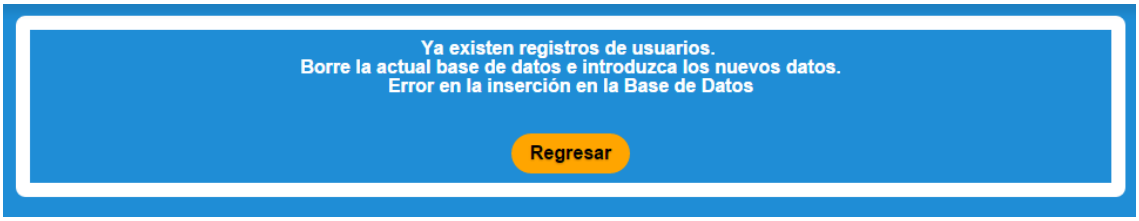

Figura 35. Muestra de error si ya hay usuarios existentes

```
$file = fopen($pathFile, "r");
if(!$file){
    echo "Ha habido un error en la apertura del fichero. <br />";
    return 0;
}else{
    while(($data = fgetcsv($file, 0, ";")) !== false){
        // Creamos un array por cada par de datos leídos
        $datos = [];
        $num = count($data);
        // Leemos de 2 en 2 datos
        for($c=0; $c < $num; $c+=2){
            for($i = 0; $i < 2; $i++){
                $datos[] = $data[$c + $i];
            }
        }
        $res = procesadoDatos($datos);
        if($res === 0){
            echo "Error en la inserción en la Base de Datos";
            fclose($file);
            unlink($pathFile);
            rmdir($pathIncludeFile);
            return 0;

        }else{
            continue;
        }
    }
}
```

Figura 36. Código que lee los datos del fichero .csv

Si el fichero .csv es correcto, se abre en modo lectura y con la función `fgetcsv` se lee sin límite hasta el final del fichero analizándolo como campos *csv*, indicando el delimitador '**;**'. La variable $`data` obtiene todos los campos de email y contraseña de los alumnos, pero en la base de datos deben ser introducidos en un solo registro por cada par email-contraseña (ver el código en Figura 36). Por eso, con otro bucle y una variable de tipo *array* se envían a otra función que procesará cada par de datos.

```
function  procesadoDatos($datos){
    $conexion = conectarDB();
    if ($conexion == null){
        return 0;
```

```
    }
    $hash = password_hash($datos[1], PASSWORD_BCRYPT);
    $sentencia = "INSERT INTO hctlab.usuarios(`email`,`hash`,`acceso`)
VALUES (?,?,0)";
    if($stmt = mysqli_prepare($conexion, $sentencia)){
        // Agregamos las variables como parámetros a la sentencia
preparada
        mysqli_stmt_bind_param($stmt, 'ss', $datos[0], $hash);
        if(!mysqli_stmt_execute($stmt)){
            echo "Ya existen registros de usuarios. <br/>";
            echo "Borre la actual base de datos e introduzca los nuevos
datos.<br />";
            return 0;
        }else{
            mysqli_stmt_close($stmt);
            mysqli_close($conexion);
            return 1;
        }
    }else{
        mysqli_stmt_close($stmt);
        mysqli_close($conexion);
        return 0;
    }
}
```

Figura 37. Función que inserta los datos leídos en un registro de la tabla usuarios

Esta función (Figura 37) primero llama a otra que se encarga de conectar con la base de datos. Después, con el formato que se estableció de en el fichero *.csv*, `$data[1]` contendrá la contraseña que se quiere asignar al alumno, de la cual se obtendrá su *hash*. Un *hash* es una huella digital relacionada a una cadena de entrada, en este caso la contraseña. Esto se logra a través de un proceso matemático de una única vía, de manera que sea muy complicado recuperar el dato original a partir de su hash. Algoritmos como MD5, SHA1 o SHA256 son vulnerables por mecanismos de fuerza bruta. PHP ofrece con la función `$password_hash` el algoritmo BCRYPT que es más seguro y al encriptar la contraseña genera una semilla (*salt*), que es un dato añadido para eliminar que el resultado pueda buscarse a partir de tablas *rainbow* (listas con códigos hash precalculados y sus datos originales). En la Figura 38 se muestra un ejemplo de *hash*.

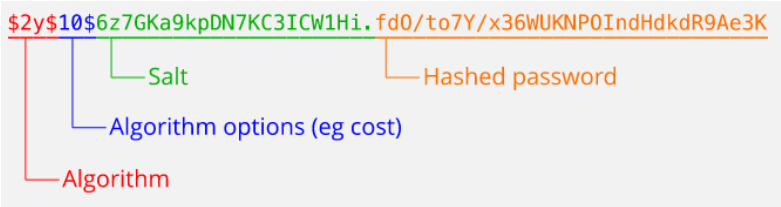

Figura 38. Ejemplo de valor devuelto por `$password_hash` [23]

Una vez almacenado el *hash* de la contraseña en la base de datos, si se quiere verificar una contraseña (por ejemplo, en la autenticación de un formulario) se hace uso de la función

$password_verify. Esta función realiza el *hash* a la variable $pass que ha sido introducida por el alumno en el formulario y si ésta y el *hash* almacenado en la base de datos coinciden, entonces se ha verificado correctamente el acceso. En la interfaz web se hacen múltiples verificaciones de esta manera según los menús accedidos.

Si la base se actualiza correctamente, se muestra la confirmación de éxito (Figura 39).

Figura 39. Mensaje tras la correcta inserción de datos

En **phpMyAdmin** podemos visualizar que se han insertado automáticamente los datos que incluían el fichero .csv mostrado en la Figura 40.

id_usuario	email	hash
6	maria.murillom@estudiante.uam.es	$2y$10$4Lwcmx8lxan9OPOAIlLRk.WOacMbyxMzCE.2QCVUEho...
7	laura.murillom@estudiante.uam.es	$2y$10$.0uQhTB/SUoVaOdEb9SZHOs6WuRgkvAQEGEGfJ2ovGg...
8	alexis.murillom@estudiante.uam.es	$2y$10$FMJWksH69jkFTeHTldjKu.Ln8YrXvwavhG42UR4stET...
9	dario.murillom@estudiante.uam.es	$2y$10$tv4CqQZvXH1fzrY.saf2ru0cHinMvy2zMtbz3OyirRy...

Figura 40. Consulta de la base a la tabla usuarios

- **borrar_BD.php:** permite vaciar la tabla de usuarios. En un formulario se pide la confirmación para realizarlo y se avisa de que una vez ejecutado no se puede recuperar. Por defecto se controla que la base de datos no se borre en caso de que se pulse el botón de *Enviar* del formulario como se observa en la Figura 41.

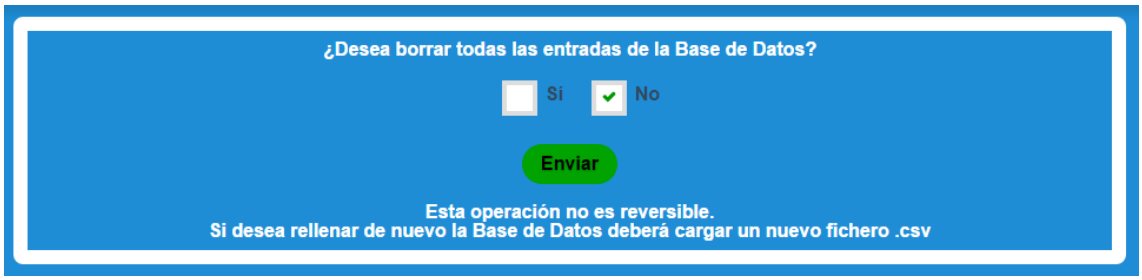

Figura 41. Formulario mostrado para borrado de la base de datos

- **borrar_BD_1.php:** esta página recoge del formulario mediante método POST la selección de borrado o no. Es importante destacar que todos los formularios que se realizan tanto en esta web como en la interfaz son realizados con el método POST que conlleva ligeramente más seguridad. El otro método llamado GET pasa las variables mediante la URL (*Uniform Resource Locator*) de la web, siendo visibles y generando una brecha de

seguridad. POST pasa los datos de forma oculta a los usuarios. Si la variable $`borrado` es 1, se debe vaciar la tabla completa.

Por otra parte, cabe destacar que la forma de ejecutar todas las consultas a la base de datos se ha realizado mediante un estilo por procedimientos (no orientado a objetos) y con sentencias preparadas o *prepared statements*. La principal ventaja de utilizar este método es la seguridad que ofrece frente a la inyección SQL (*SQL injection*), una vulnerabilidad conocida del código *SQL*. Esta vulnerabilidad consiste en la inyección de código para que se infiltre dentro de los parámetros de consulta y se ejecute dentro del servidor de la base de datos, pudiendo acceder incluso al resto del servidor.

```
$sentencia = "SELECT id_usuario FROM usuarios WHERE `email`=
              '$email_formulario' AND `hash`= '$pass_formulario'";
$resultado = mysqli_query($conexion, $sentencia);
```

Figura 42. Ejemplo de consulta a MySQL tradicional

En el ejemplo de la Figura 42, un intruso podría insertar a través de un formulario una sentencia que se evalúa a verdadero siempre, sin necesidad de introducir el dato verdadero. Se ha probado un ejemplo en la base de datos como muestra la Figura 43.

Figura 43. Ejemplo de inyección SQL

Como se puede ver, se ha realizado una consulta que debería ser falsa, pues no se ha introducido el email de un usuario que esté en la base de datos ni una contraseña. Sin embargo, el intruso al añadir la evaluación siempre verdadera de *'or '1'='1* entraría al sistema sin tener usuario. Con las sentencias preparadas los datos no se añaden en la consulta, sino que se marcan los espacios donde irán los parámetros con un signo de interrogación ($`mysqli_stmt_prepare`) y después se ligan las variables a los espacios marcados. La Figura 44 muestra un ejemplo de sentencia preparada.

```
$sentencia = "SELECT id_reserva FROM reserva WHERE `email`=?";
if($stmt = mysqli_prepare($conexion, $sentencia)){
```

```
    // Agregamos la variable $email como parámetro a la sentencia
preparada
    mysqli_stmt_bind_param($stmt, 's', $email);
    if(!mysqli_stmt_execute($stmt)){
        echo "Error en la Base de Datos <br />";
        mysqli_close($conexion);
    }else{
        // Almacenamos el resultado
        mysqli_stmt_store_result($stmt);
        //Procesar
    }
}
```

Figura 44. Ejemplo de sentencia preparada para evitar inyección SQL

Al mandarse la consulta sin datos al servidor y enviárselos después de que se sepan los parámetros que se esperan, se evita que un atacante pueda introducir su código en las consultas. Todas las consultas que se realizan en esta interfaz web han sido construidas de esta forma para mayor seguridad del sistema.

4.4 Interfaz web

A continuación se analizará la estructura y funcionamiento de la interfaz desarrollada. La Figura 45 y Figura 46 detallan el esquema de páginas web que forman la interfaz y el contenido del directorio completo.

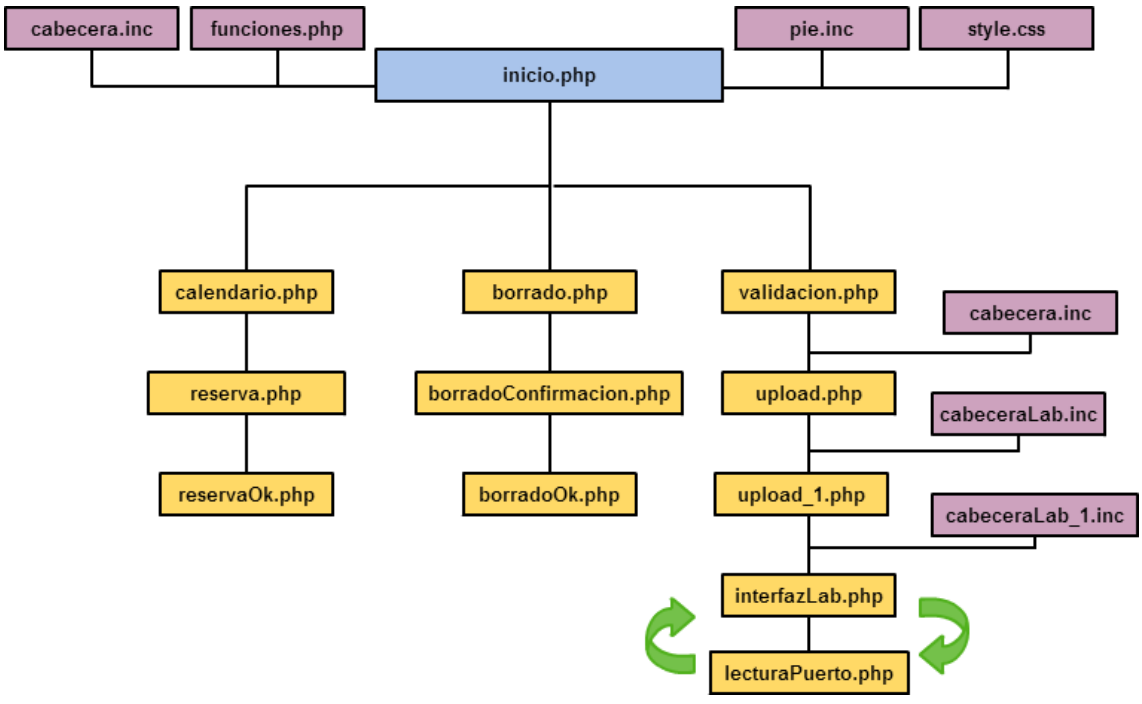

Figura 45. Esquema de la relación de páginas de la interfaz web

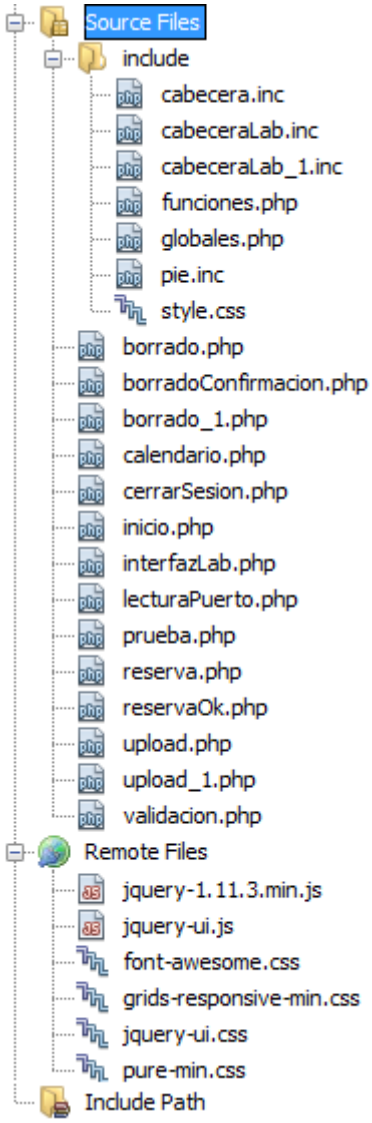

Figura 46. Directorio de la interfaz web

- **inicio.php:** página principal de acceso para la interfaz (Figura 47)

Figura 47. Captura de la página principal de la interfaz

- **calendario.php:** se accede mediante el menú de la cabecera y es la primera página de la secuencia para reservar una hora de acceso a la interfaz (Figura 48).

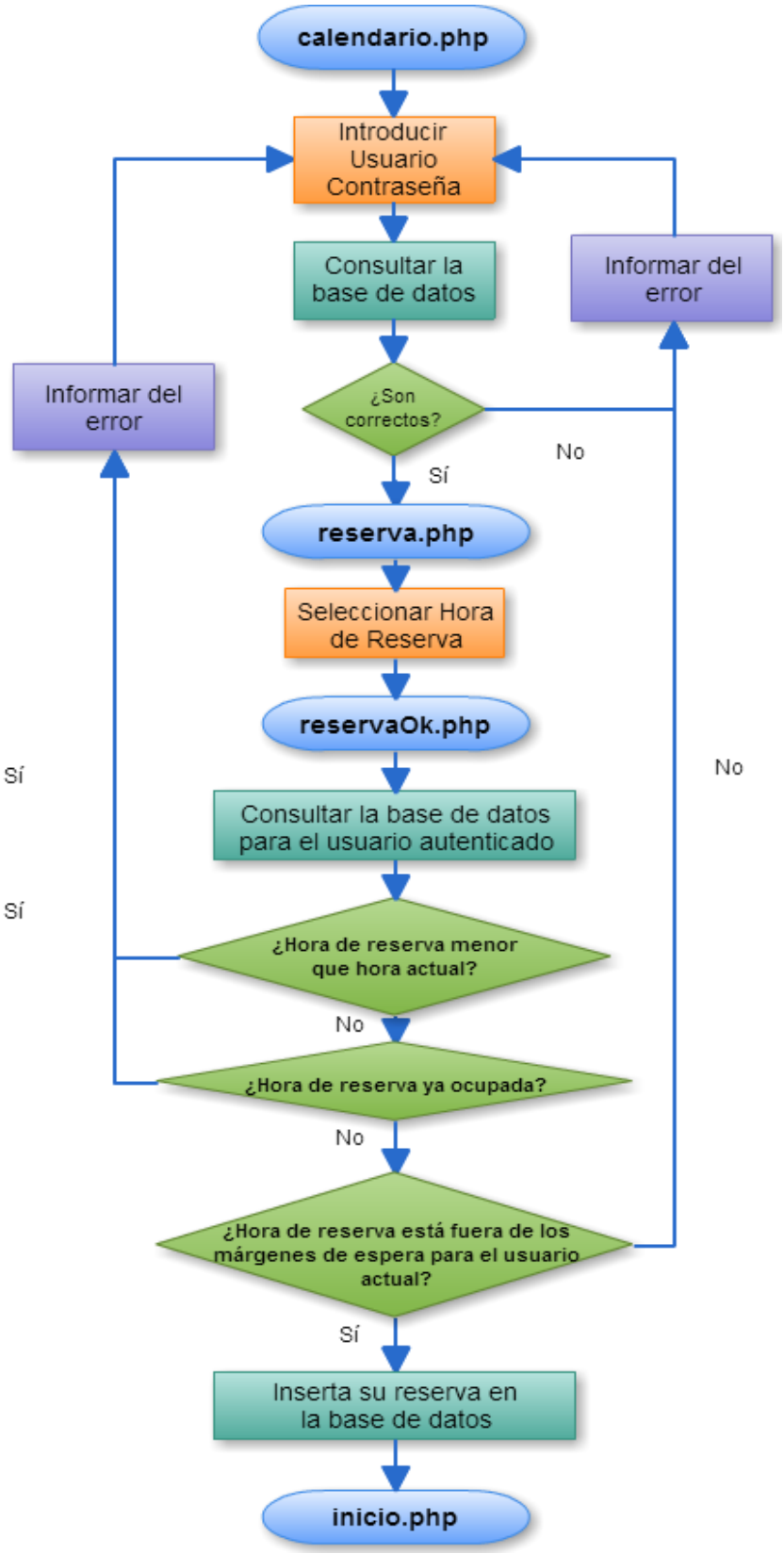

Figura 48. Diagrama de flujo para la reserva de horarios

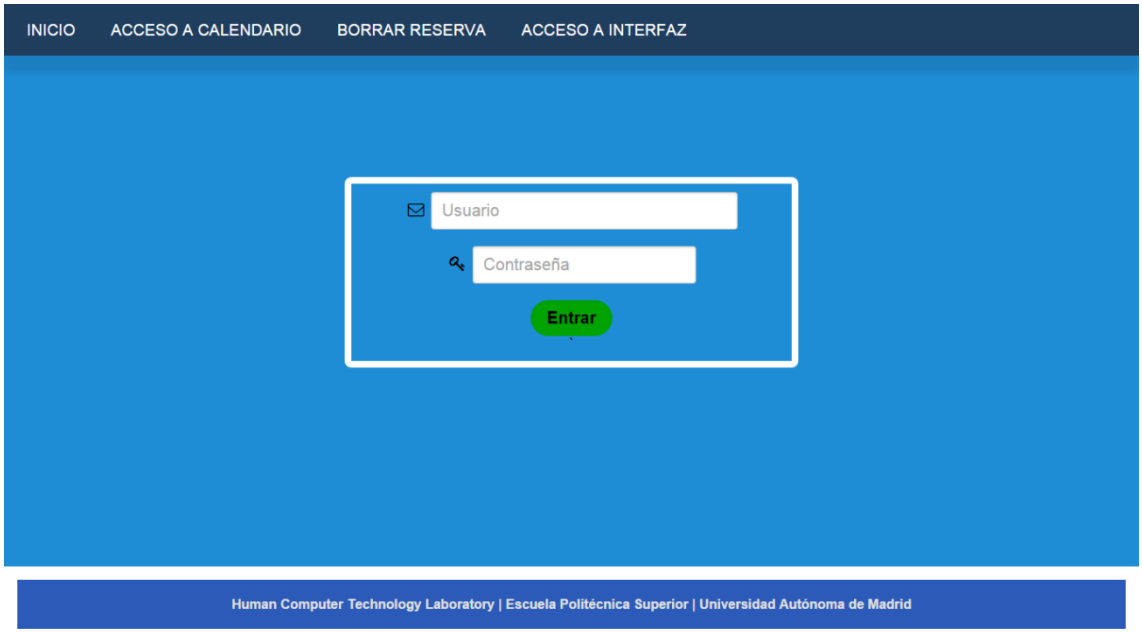

Figura 49. Captura del formulario para acceder a reservar hora

El usuario debe autenticarse para acceder a la zona de reservas como muestra la Figura 49. En código PHP se han controlado los posibles casos erróneos que se pueden dar, que serán comprobados en cada zona que requiera autenticación como ocurrirá en posteriores secciones:

- No se han introducido datos en el formulario.
- Se ha introducido solamente el usuario (email).
- Se ha introducido solamente la contraseña.
- Se ha introducido usuario y contraseña pero uno de los datos o ambos son incorrectos.

- **reserva.php:** es la página que gestionará los datos recibidos del formulario. Si no ocurre ninguno de los errores mencionados, se mostrarán los menús desplegables que permiten escoger la hora de reserva. Como métodos de seguridad, ya se comentó que para todos los formularios se utilizaría el método POST y en el caso de autenticación, además, debemos filtrar los datos que recibimos del usuario.

```
// Validamos los datos recibidos del formulario
$usuario_calendario = filter_input(INPUT_POST, 'emailreserva',
FILTER_VALIDATE_EMAIL);
$pass_calendario    = filter_input(INPUT_POST, 'passreserva',
FILTER_VALIDATE_INT,
                        ['min_range' => 1, 'max_range' => MAX_NUM_PASS]);
```

Figura 50. Código que filtra los datos recibidos del formulario

Con el código anterior (Figura 50), las funciones `filter_input` están preparadas para filtrar los datos recibidos en este caso mediante POST, ya que podrían recibirse caracteres no esperados para el correcto funcionamiento de la web. En el primer caso validamos que sea un email correcto y en el segundo, que la contraseña sea un número `int` y que se encuentre dentro del rango especificado (`MAX_NUM_PASS` se ha declarado como constante, ver Anexo I.3 si se quiere modificar este parámetro).

```
if(isset($usuario_calendario) && isset($pass_calendario) &&
    preg_match('/(@estudiante.uam.es)/', $usuario_calendario)){
```

Figura 51. Código para comprobación de usuario y contraseña

Una vez validados los datos se comprueban (como muestra la Figura 51) que han sido asignados y que además el email del usuario cumple con el patrón de correo institucional de la UAM (función `preg_match`). Si ocurre algún error en todo este proceso, se muestran mensajes como los de la Figura 52:

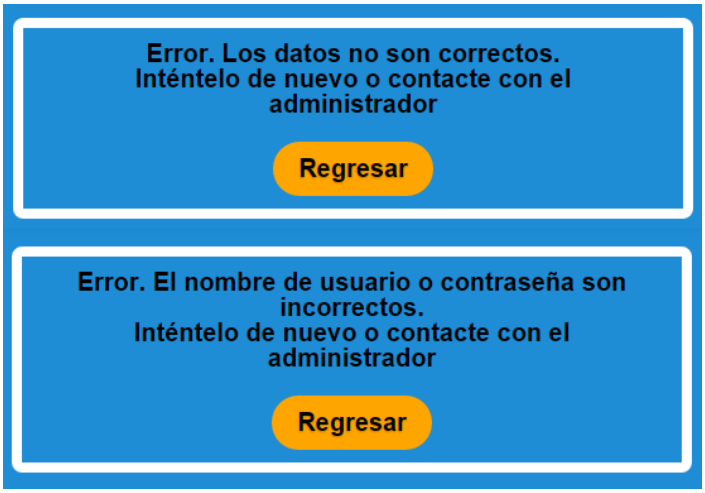

Figura 52. Información de errores al usuario en reserva.php

Si no ha habido errores, se muestran los menús desplegables para seleccionar la hora de reserva deseada como en la Figura 53.

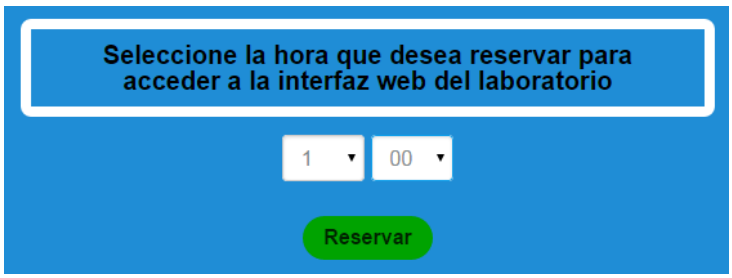

Figura 53. Menús desplegables para reserva de hora

- **reservaOk.php:** una vez que se ha escogido la hora para la reserva, esta página gestiona tres posibles conflictos:

- Que la hora de reserva no sea menor que la hora actual.
- Que el usuario tenga realizada una reserva anteriormente y haya seleccionado una hora dentro del tiempo de espera para una nueva reserva.
- Que la hora que ha seleccionado no esté ocupada por otro usuario.

De nuevo, se validan los datos recibidos por el formulario y estas comprobaciones se realizan consultando a la base de datos. En caso de cumplirse algún conflicto informan del error y permiten regresar a la página de **calendario.php** como se mostraba en el diagrama de flujo de la. Si no hay errores (Figura 54), informan del éxito de la reserva (Figura 55).

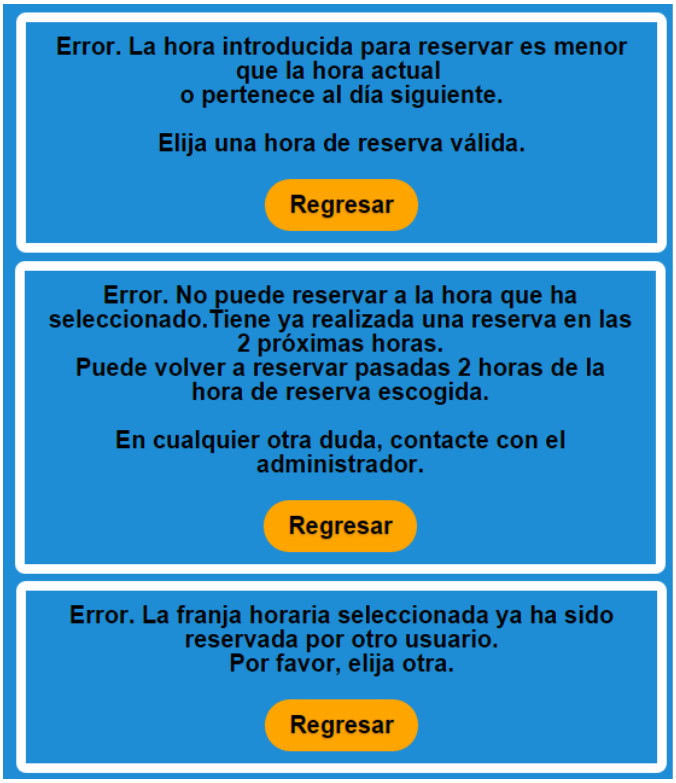

Figura 54. Información de errores al usuario en reservaOk.php

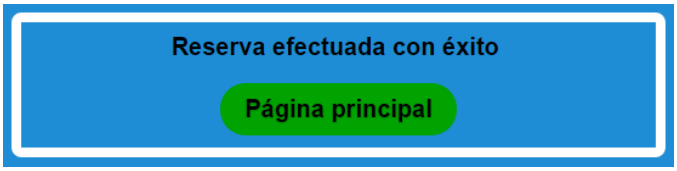

Figura 55. Información de reserva exitosa

- **borrado.php:** tiene el mismo formato que la página de **calendario.php** (ver Figura 56). Permite iniciar el proceso para borrar la reserva que tiene almacenada un usuario autenticado.

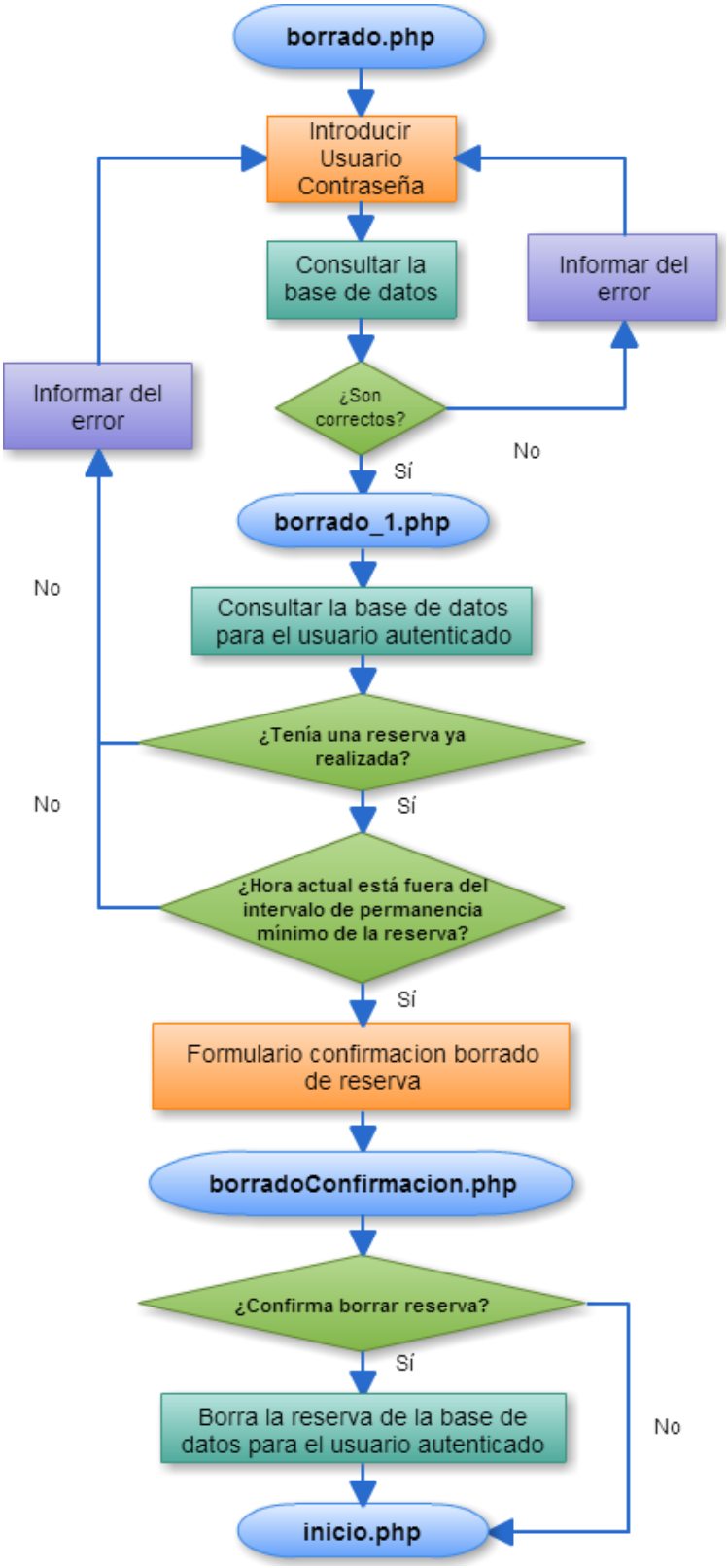

Figura 56. Diagrama de flujo para borrar reservas

- **borrado_1.php:** se controlan dos posibles conflictos que pueden suceder para no borrar una reserva:

- El primero es que el usuario que ha accedido no tenga una reserva realizada.
- El segundo, que sí tenga reserva pero la hora a la que se quiere borrar la reserva se encuentre en el intervalo de permanencia de la reserva.

Si se da alguno de los casos citados o el usuario no se ha autenticado correctamente se muestran errores como los siguientes (Figura 57):

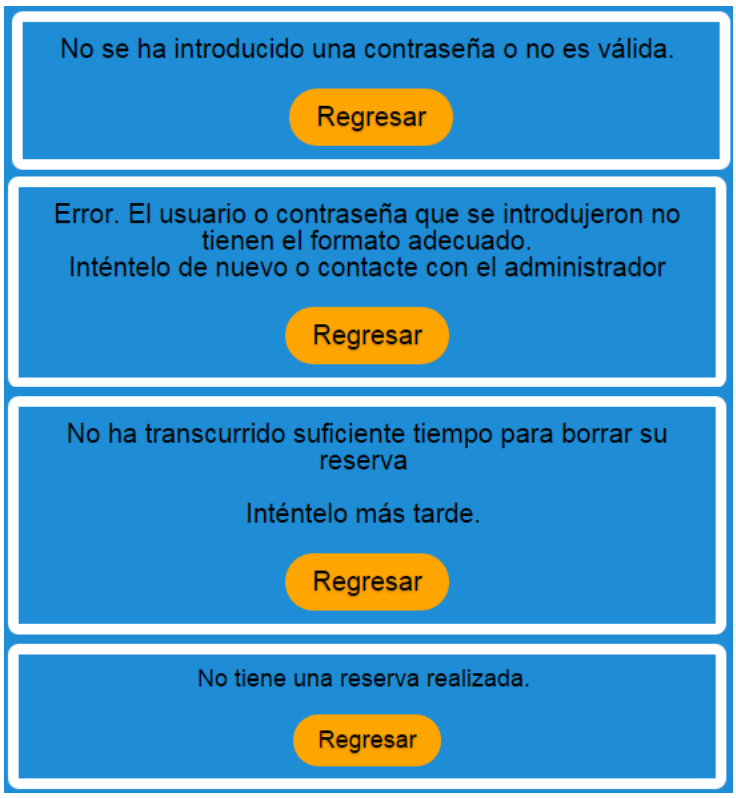

Figura 57. Información de errores al usuario en borrado_1.php

Si no hay errores, se accede al formulario para confirmar el borrado. Éste muestra la hora de la reserva o reservas al usuario como la siguiente Figura 58:

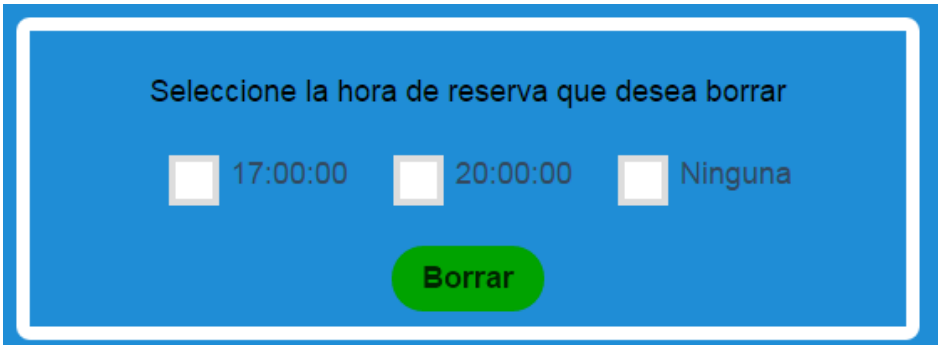

Figura 58. Formulario para selección de borrar reserva

Si se decide no borrar la reserva se muestra el correspondiente aviso (Figura 59) y devuelve a la página de inicio. Si se decide borrar, se accede a la base de datos y se borra el registro correspondiente al usuario que accedió.

Por último, se informa del éxito de borrado (Figura 60). Todo ello en la página **borradoConfirmacion.php**.

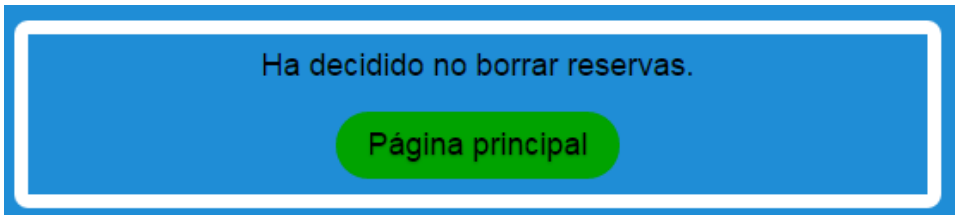

Figura 59. Información de decisión no borrar reserva en borradoConfirmacion.php

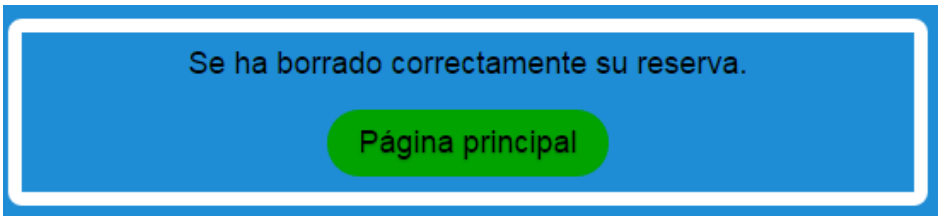

Figura 60. Información de éxito en borrado de reserva de borradoConfirmacion.php

- **validacion.php:** también con el mismo formato que se indicó en **calendario.php** y **borrado.php**. Permite iniciar el acceso a la interfaz web para programar la FPGA. El diagrama de flujo se muestra a continuación en la Figura 61.

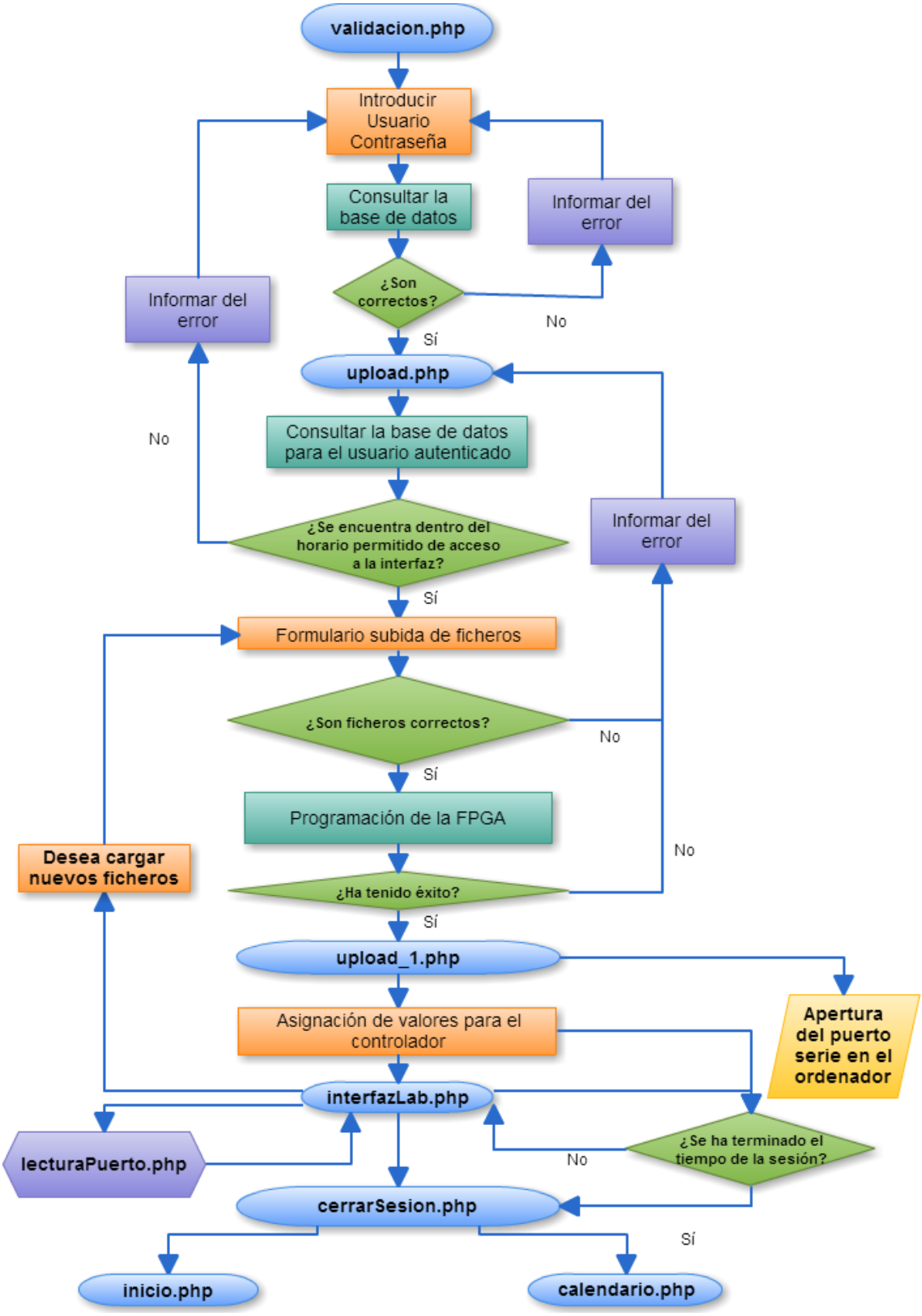

Figura 61. Diagrama de flujo para la interfaz del laboratorio

- **upload.php:** cuando un usuario accede por primera vez, se inicia una sesión para la interfaz. El acceso de los usuarios a esta página puede suceder de dos formas diferentes:

- La tradicional mediante el formulario, autenticación y validación de los datos introducidos.
- Tras elegir la nueva opción de la cabecera subir nuevos ficheros una vez que ha accedido a la interfaz.

La última opción se verifica con el valor asignado a una variable de estado en la sesión iniciada. Si verifica el estado que le corresponde y que el tiempo de uso de la interfaz no se ha excedido, permite mostrar el siguiente formulario de la Figura 62:

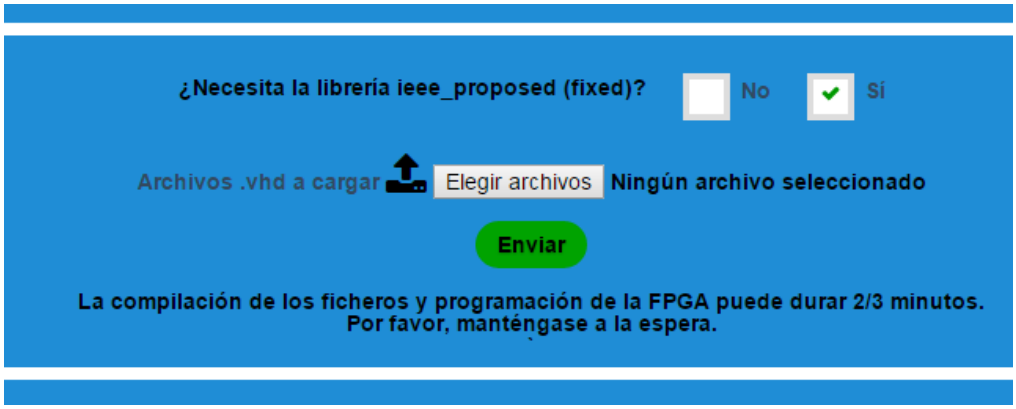

Figura 62. Formulario de subida de archivos 'upload.php'

El uso de sesiones para acceso a la interfaz viene motivado por la necesidad de conservar información de un usuario (id, estado, nombre de usuario, tiempo de reserva,…) al pasar de una página a otra. Con su uso, los datos se almacenan en el servidor de manera que no pueden ser vistos o editados por el cliente por lo que la seguridad no se compromete, a diferencia del uso de *cookies.*

Una vez validados los datos del usuario y comprobando con consultas a la base de datos que tiene acceso y es la hora de uso asignada para la interfaz, se registra la sesión con los siguientes datos (Figura 63):

```
$_SESSION['usuario'] = $email;
$_SESSION['pass'] = $pass;
$_SESSION['hora_reserva'] = $hora_reserva_lab;
$_SESSION['timeout'] = $hora_reserva_lab + $t_interfaz;
$ SESSION['estado'] = 1;
$_SESSION['id'] = session_id();
$limite_sesion = $_SESSION['timeout'] - horaActual();
$limite_sesion_ms = $limite_sesion * 1000;
```

Figura 63. Datos para el registro de sesión en 'upload.php'

Las dos últimas asignaciones a variables se utilizan para controlar el tiempo transcurrido en la sesión, a modo de contador descendente. Cuando llegue a 0, significa que la hora máxima de uso de la interfaz se ha alcanzado y se debe cerrar sesión. Este control,

además de hacerse en el código PHP, se ejecuta en cada página con un script que se ejecuta en el cliente, utilizando JavaScript (Figura 64):

```
<script type= "text/javascript">
    // Captura de la variable que en el código PHP calculaba los ms
restantes
    var milisegundos = <?php echo $limite_sesion_ms;?>;
    // Si se ha alcanzado el límite, cerramos sesión automáticamente
    // invocando la página de cerrarsesion.php
    function killerSession(){
        setTimeout("window.open('cerrarsesion.php','_top');",
milisegundos);
    }
    // Llamada a la función para cerrar la sesión después de que toda la
página se ha cargado
    window.onload = function(){
        killerSession();
    };
</script>
```

Figura 64. Código en *JavaScript* para controlar el uso de la interfaz web

A lo largo de la página, si ha ocurrido algún error se mostrarán al usuario los posibles avisos que se muestran a continuación en la Figura 65:

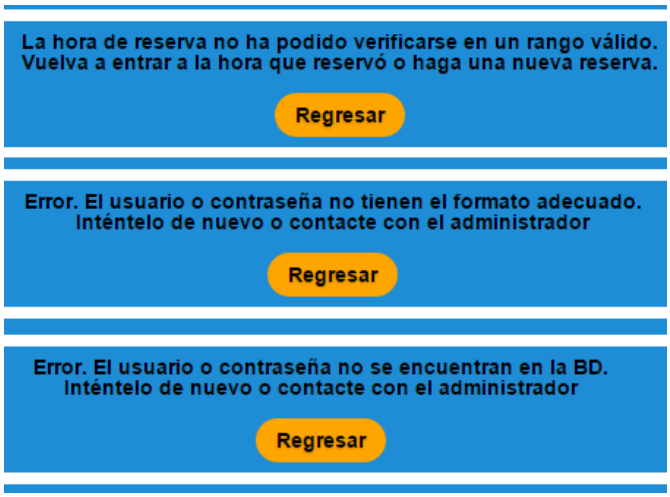

Figura 65. Información de errores al usuario en 'upload.php'

- **upload_1.php:** en esta página se realiza el manejo de los datos del formulario anterior. Primero se asigna a una variable el valor de si se decidió utilizar la librería *ieee_proposed* o no (para el caso de este laboratorio sí es necesaria, pero en otros diseños no se debe utilizar para compilar los .vhd, ya que generaría errores). Después se llama a la función `borrarDirAlumno()` para eliminar, en caso de que existiese, el directorio asociado a ese alumno que impediría la subida de ficheros. A continuación se llama a la función `manejoArchivo($libreria)`. Esta función retornará un valor de tipo *int* que se utilizará para saber si tuvo éxito y mostrar la interfaz o no. Su funcionamiento se divide en varias fases:

Comprobación de archivos: obtiene mediante la función `asignacionRutas()` los directorios que necesita para procesar los archivos. Esta función devuelve un *array* y se listan de forma adecuada. Es importante que desde el servidor se hayan configurado correctamente como se explica en el Anexo I.3.

Dado que se reciben varios archivos con extensión .vhd (en la variable superglobal `$_FILES)`, se crearán *arrays* para mantener referencias a cada uno. En un bucle *for*, para cada uno de los archivos se realizan comprobaciones de seguridad tales como:

- Obtención del nombre del archivo con la función *basename* que devuelve nombre y extensión del archivo extrayendo posibles rutas del archivo e impidiendo ataques de directorio traversal o *traversal path* en inglés (vulnerabilidad informática con la que se podría acceder a directorios y archivos del servidor).
- Comprobación de única extensión permitida en los archivos subidos como .vhd
- Rechazo de ficheros con otras extensiones como '.exe' o '.php5'
- Comprobación de que sólo incluye caracteres alfanuméricos, '_' ó '-'
- Comprobación de los errores (no se ha subido archivo, tamaño excedido, error en escritura, …)

Los posibles errores que se muestran en pantalla al usuario son (Figura 66):

Figura 66. Información de errores al usuario en Comprobación de archivos 'upload_1.php'

Cada fichero que pase con éxito todas las comprobaciones, será copiado a la carpeta generada para los siguientes pasos de procesado.

Compilación de archivos: cada archivo correctamente copiado, será compilado por ModelSim llamando a la función `compilacionVHD()` y los argumentos necesarios para las rutas de directorios y ficheros. Es aquí donde se tiene en cuenta la elección de la librería. Se genera un *array* que contiene los comandos a ejecutar invocando a ModelSim. El comando *'vlib'* crea la librería con nombre el nombre que se especifica y *'vcom'* compila el archivo .vhd con la librería que se le indica. En el caso de estas prácticas, también se compilan los ficheros necesarios de la librería *ieee_proposed*,

```
if($libreria == 0){
    $path[] = 'vlib work';
    $path[] = 'vcom -reportprogress 300 -work work ' . $pathFile;
}elseif($libreria == 1){
    $path[] = 'vlib work';
    $path[] = 'vlib ieee_proposed';
    $path[] = 'vcom -reportprogress 300 -work ieee_proposed ' .
            $pathLib_remota . 'fixed_float_types_c.vhdl';
    $path[] = 'vcom -reportprogress 300 -work ieee_proposed ' .
            $pathLib_remota . 'fixed_pkg_c.vhdl';
    $path[] = 'vcom -reportprogress 300 -work ieee_proposed ' .
            $pathLib_remota . 'fixed_synth_c.vhdl';
    $path[] = 'vcom -reportprogress 300 -work work ' . $pathFile;
}else{
    echo "Ha habido un error en la compilación y las librerías.";
    $i = 0;
    return $i;
}
```

Figura 67. Código PHP para compilación de archivos .php en línea de comandos

Una vez construido el *array* con la librería y ruta de archivos que debe compilar tal y como muestra la Figura 67, se ejecuta cada elemento del array con la función *system( )* que permite ejecutar programas externos como línea de comandos en Windows (*cmd.exe*). A su vez, se vuelca la salida a un fichero de texto llamado *compilación.txt*. Este fichero es leído en busca de líneas donde se muestre la palabra 'Error:' para poder mostrarlo al usuario en caso de que hubiese errores de compilación. También se ha hecho un exhaustivo análisis de las cadenas de error para no mostrar al usuario las rutas del directorio del archivo que se compila por la web. La Figura 68 ilustra un ejemplo.

Figura 68. Información de errores en la compilacion de archivos de 'upload_1.php'

Por cada archivo se controla en una variable de tipo *array* a modo de *'flag'* si hubo o no errores. Si se generase algún error, internamente se borrarían los ficheros y directorios que se crearon en todo el proceso.

Generación archivo .bit: en este caso debemos generarlo mediante línea de comandos. Los procesos que sigue Xilinx ISE para generar un fichero .bit y programar la FPGA con los archivos que recibe son los siguientes de la Figura 69:

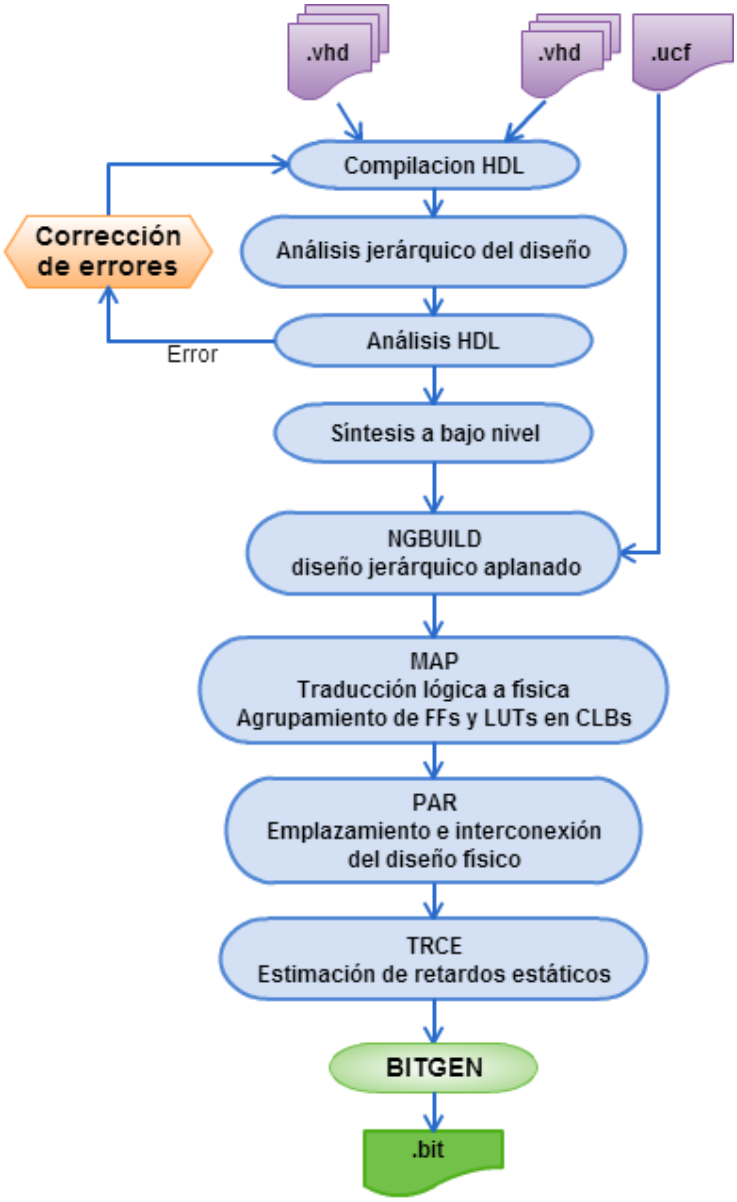

Figura 69. Esquema de generación de archivo bitstream (.bit)

Para poder implementar todos estos procesos de forma automática necesitamos generar un archivo que contenga todos aquellos requeridos para el diseño. En la función `generacionBit()` se crea y escribe en la carpeta del usuario un archivo de texto nombrado '*top.prj*' con los ficheros que nos ha suministrado el alumno más los que debemos proporcionar con los módulos UART, el top.vhd y librerías. En el archivo creado (ver Figura 70) se escriben con el formato adecuado mediante código PHP todos los ficheros, controlando que los que proporcionamos al usuario no coinciden en nombre con los suyos. En tal caso, se renombra el fichero copiado que proporcionamos para evitar problemas en el diseño. El formato que contendrá el archivo de texto creado será similar al mostrado a continuación en la en el caso de estas prácticas.

```
vhdl ieee_proposed "D:\Prueba_TFG\ieee_proposed\fixed_float_types_c.vhdl"
vhdl ieee_proposed "D:\Prueba_TFG\ieee_proposed\fixed_pkg_c.vhdl"
vhdl ieee_proposed "D:\Prueba_TFG\ieee_proposed\fixed_synth_c.vhdl"
vhdl work "Controlador_2014.vhd"
vhdl work "GestorADC.vhd"
vhdl work "GestorPWM.vhd"
vhdl work "Regulador.vhd"
vhdl work "bbfifo_16x8.vhd"
vhdl work "kcuart_rx.vhd"
vhdl work "kcuart_tx.vhd"
vhdl work "top_1.vhd"
vhdl work "uart_rx.vhd"
vhdl work "uart_tx.vhd"
```

Figura 70. Formato de fichero top.prj

El siguiente paso para generar el *.bit* es ejecutar el programa *XFLOW* que provee Xilinx. Nos permite automatizar la síntesis, implementación y flujos de simulación mostrados en la especificando con archivos de opciones lo que queremos. En este caso son:

- **–p xc3s1000ft256-4**: especificamos el dispositivo, encapsulado y velocidad de la FPGA utilizada.
- **–implement balanced.opt**: se invoca a un archivo de flujo para ejecutar los procesos de NGBUILD, MAP, PAR y TRACE. En este caso, se elige balanced.opt que mantiene un equilibrio entre la velocidad con la que se ejecutarán y el rendimiento.
- **–config bitgen.opt**: se especifica que también debe llamar al programa BITGEN para crear el archivo *.bit*
- **–wd** : especificamos el directorio donde copiará los archivos de flujo, ejecutará todos los programas y generará todos los archivos necesarios para el *.bit*
- **–synth xst_vhdl.opt top.prj**: sintetiza el diseño para implementarlo en una FPGA con el archivo de flujo indicado. Éste permite optimizar en velocidad los archivos fuente VHDL descritos en *top.prj* para incrementar la velocidad del diseño.

Todas esas opciones se ejecutan de nuevo con *system( )* y se vuelca la salida de la ejecución a dos archivos *.txt*, uno contendrá posibles errores generados y el otro todos los comandos y descripciones ejecutadas por *XFLOW*. De éste último archivo, podremos leer si hubo éxito o no en la generación del *.bit* (‘*xflow done!*’).

Programación de FPGA con .bit: se realiza con el programa *iMPACT* de Xilinx. Permite descargar un fichero *.bit* en la FPGA (a través del cable JTAG conectado entre PC-FPGA) y programarla con la información contenida en él. *iMPACT* posee un método para ejecutarse desde línea de comandos llamado *Batch mode*, de forma que puede leer y ejecutar el contenido de un fichero *.cmd*.

En el código PHP mediante la función `generacionComandosImpact()`, se debe crear un fichero *comandos.cmd* con las acciones a ejecutar para detectar el cable JTAG en la FPGA, asignar el archivo *.bit* y programarla (ver Figura 71).

```
function generacionComandosImpact($pathnameVHD){
    $pathCMD= $pathnameVHD . 'comandos.cmd';
    $file = fopen($pathCMD, "w+");
    if($file === false){
        echo "Error al crear fichero de comandos. Regrese al inicio";
        return 0;
    }else{
        $comandos = [];
        $comandos[] = 'setMode -bs';
        $comandos[] = 'setCable -p auto';
        $comandos[] = 'identify';
        $comandos[] = 'assignFile -p 1 -file ' . '"' . $pathnameVHD .
'top.bit' . '"';
        $comandos[] = 'program -p 1';
        // Escribimos los comandos
        for($i = 0; $i < count($comandos); $i++){
            if(!fwrite($file, $comandos[$i] ."\r\n")){
                echo "Error al generar comandos para programar la FPGA.";
                return 0;
            }
        }
    }
    fclose($file);
    return 1;
}
```

Figura 71. Función para comandos de programa iMPACT

Se ejecuta el comando *impact –batch "comandos.cmd"* con *system( )* y volcando a otros dos ficheros como anteriormente, se leen los errores o el éxito de la programación.

```
$impactError_1 = preg_match('/Cable connection failed/', $line_2);
$impactError_2 = preg_match('/Cable autodetection failed/', $line_2);
$impactDone = preg_match('/Programmed successfully/', $line_2);
```

Figura 72. Comandos para detección de errores de ejecución del programa iMPACT

Cualquier error detectado (Figura 72), en este proceso informa al usuario y permite regresar a la página de **'upload.php'** (Figura 73).

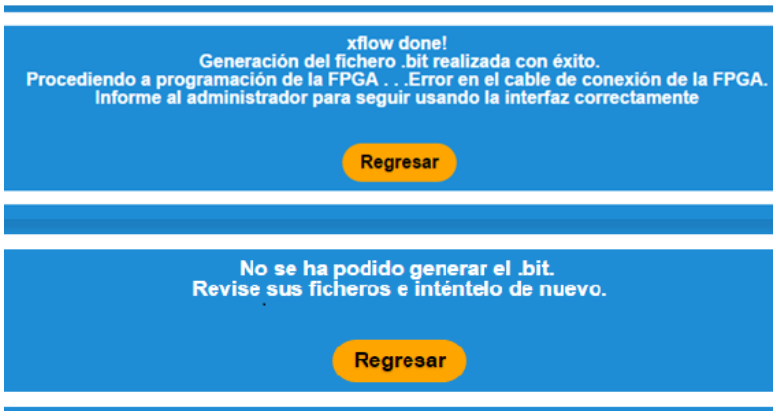

Figura 73. Información de errores al usuario en 'upload_1.php'

Si no hay errores, se permite asignar los valores iniciales a la FPGA de acuerdo al funcionamiento de las prácticas como en la Figura 74:

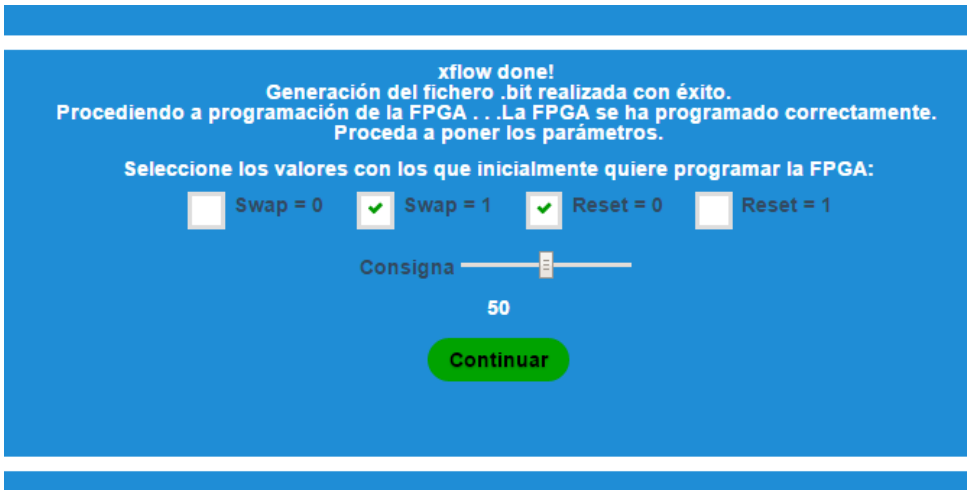

Figura 74. Formulario para valores iniciales del Controlador en 'upload_1.php'

- **interfazLab.php:** muestra la interfaz del osciloscopio, los valores que se pueden elegir para asignar de nuevo al controlador y el valor de la lectura del puerto serie. Cada vez que el alumno pulse el botón *Enviar*, los 3 datos se enviarán por el puerto serie y gracias a la conversión USB-UART, llegarán hasta la FPGA donde se asignarán los nuevos valores al controlador como se explicó en el 3.1.4 Módulo de comunicación UART. Como se comentó en el apartado 3, se hace uso de AJAX para mostrar el valor leído del puerto sin tener que recargar la página entera, mostrándose el valor actualizado cada 5 segundos. La petición AJAX (Figura 75) se realiza a la página **lecturaPuerto.php** que ejecuta el programa que leía el valor del puerto y dicho valor se imprime por pantalla mostrándolo al usuario en su valor decimal y binario.

```
<script>
function rellenaInfo() {
    $.ajax({
            url: "/labRemoto/lecturaPuerto.php",
            dataType: "html"
        }).done(function(msg) {
            // Método que nos permite acceder a la referencia del id
mensaje,
            // capturando los datos de los LEDs
            document.getElementById("mensaje").innerHTML=msg;
        });
}
    loop = setInterval(rellenaInfo,5000);
</script>
```

Figura 75. Petición AJAX para lectura del puerto serie

Con la interfaz del osciloscopio integrada, se puede acceder a diferentes menús para visualizar el osciloscopio. Uno de ellos es el de capturar una imagen del osciloscopio como muestra la Figura 76:

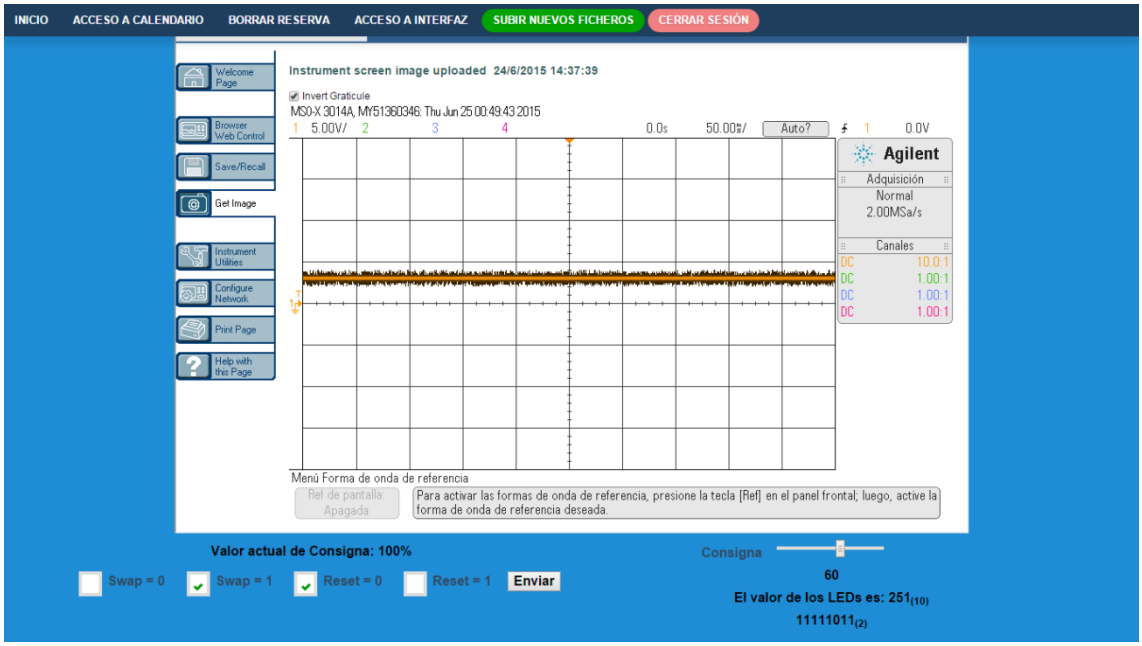

Figura 76.Interfaz del laboratorio remoto completa (I)

Finalmente, siempre que Java esté activado y correctamente configurado en el ordenador del usuario otro menú permite lanzar una aplicación para un manejo más avanzado del osciloscopio como se muestra finalmente en la Figura 77:

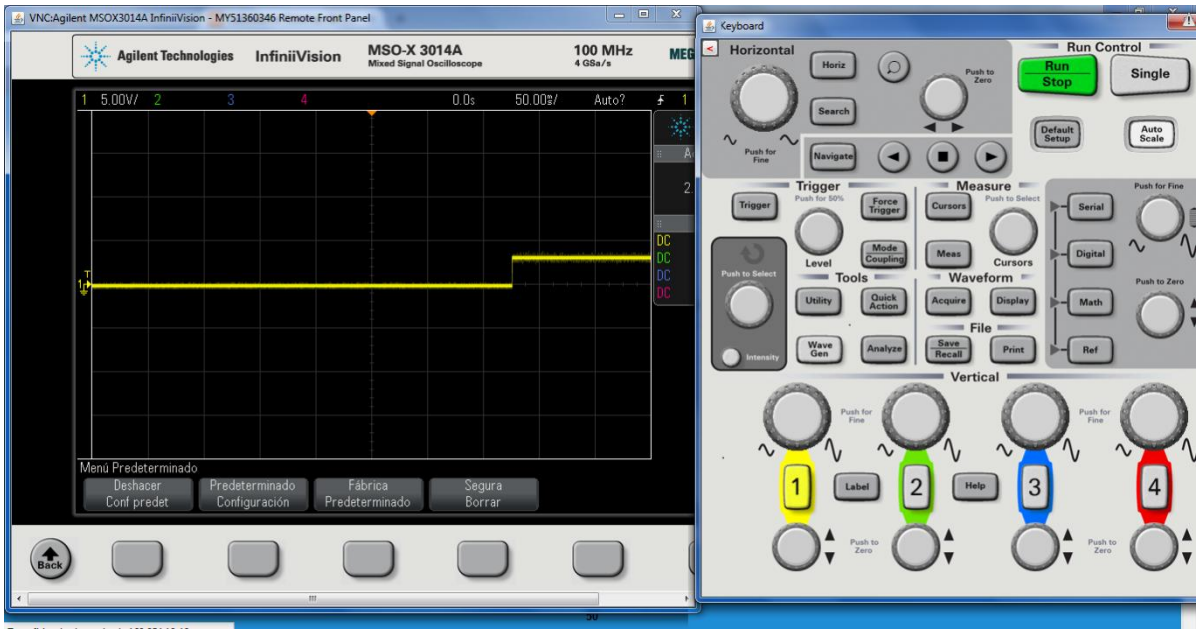

Figura 77. Interfaz del laboratorio remoto completa (II)

4.4.1 Resultados

Una vez analizado el funcionamiento de cada uno de los componentes del laboratorio remoto, cabe concluir que se han implementado varios reguladores con las dos plantas disponibles para estas prácticas y han funcionado correctamente.

Asimismo, se ha realizado una batería de pruebas extensa para comprobar que los errores de compilación en los ficheros de los alumnos se muestren adecuadamente. También se ha probado la subida de diferentes archivos al servidor que no fuesen acordes al funcionamiento de las prácticas para verificar que se rechazaban y mostraban los correspondientes errores que se han analizado anteriormente.

Además, se ha probado el correcto funcionamiento de la base de datos, con múltiples reservas, usuarios y diferentes horarios, así como la actualización de la base con el sistema de gestión mediante ficheros *.csv*. También se han realizado pruebas para verificar la protección frente a inyección SQL.

Por último, se ha comprobado que la visualización del laboratorio remoto es correcta en diferentes navegadores de Internet.

5. CONCLUSIONES

El continuo desarrollo de las TIC nos ofrece grandes beneficios a lo largo de nuestro día a día. En el ámbito educativo, las universidades han empezado a desarrollar laboratorios virtuales y remotos en beneficio de los alumnos que permite la independencia de los laboratorios tradicionales al poder realizar prácticas en cualquier horario y lugar.

En el caso de este TFG, se ha realizado un laboratorio remoto para la asignatura de Sistemas de Control del Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación de la Universidad Autónoma de Madrid, a partir de la base conocida de su laboratorio tradicional. El proyecto se inició investigando las tecnologías y dispositivos de los que se debían disponer, así como la elección de software y lenguajes de desarrollo que más se adecuaban para el propósito. Una vez concretados, se fueron integrando los módulos y comprobando su correcto funcionamiento por separado hasta la integración final.

En la integración final se ha demostrado que la interfaz web funciona de acuerdo a los objetivos planteados al principio, añadiendo dos funcionalidades: la primera es la gestión de la base de datos mediante subida de ficheros .csv para su actualización y la segunda es la generación del archivo .bit a partir de archivos .vhd.

El servidor se ha desplegado en un equipo en el grupo de investigación HCTLab y está preparado para su funcionamiento en el próximo curso académico de la citada asignatura.

5.1 Líneas futuras

Como líneas de desarrollo futuro, este TFG podría ampliarse con más módulos de FPGA, UART y circuitos electrónicos analógicos para dar mayor servicio a los alumnos o si fuese requerido por el código, añadir nuevas librerías y otros parámetros de la UART. Para añadir mayor seguridad también podría implementarse un *proxy* entre el cliente y el osciloscopio de forma que el osciloscopio únicamente esté conectado al *proxy* a través de una ip privada, y dicho *proxy* redirija los datos del osciloscopio al cliente. Además, este TFG sirve de base para la implementación de otros laboratorios que requieran el uso de programación e interacción remota de FPGAs, puesto que se han desarrollado numerosas subrutinas necesarias para ello.

BIBLIOGRAFÍA

[1] Duarte, M., y Brian P., "*The Virtual Laboratory for the Disabled*," Frontiers in Education Conference, 2001. 31st Annual , vol.3, no., pp.S1C,23-6 vol.3, 2001 Butz.

[2] Zamora Musa, Ronald., "*Laboratorios Remotos: Actualidad y Tendencias Futuras*," Scientia et Technica, [S.l.], v. 2, n. 51, p. 113-118,ago. 2012. ISSN 0122-1701.

[3] Koretsky, M.D., D. Amatore, C. Barnes, y S., "*Enhancement of Student Learning in Experimental Design Using a Virtual Laboratory*," Education, IEEE Transactions on , vol.51, no.1, pp.76,85, Feb. 2008. Kimura.

[4] P. J. Mosterman, M. A. M. Dorlandt, J. O. Campbell, C. Burow, R. Bouw, A. J. Brodersen, and J. Bourne, "*Virtual engineering laboratories: Design and experiments*," J. Eng. Educ., vol. 83, pp. 279–285, Jul. 1994.

[5] Uran, S., y K., "*Virtual Laboratory for Creative Control Design Experiments*," Education, IEEE Transactions on , vol.51, no.1, pp.69,75, Feb. 2008 Jezernik.

[6] Alberto Sánchez González, "*Aportaciones mediante implementación basada en sistemas embebidos al control digital de convertidores conmutados*". Tesis doctoral, Universidad Autónoma de Madrid, 2013.

[7] Chang, Gao-Wei, Zong-Mu Yeh, Hsiu-Ming Chang, y "*Teaching photonics laboratory using remote-control web technologies*," Education, IEEE Transactions on , vol.48, no.4, pp.642,651, Nov. 2005 Shih-Yao Pan.

[8] H. Vargas, J. Sanchez , C. A. Jara , F. A. Candelas , F. Torres and S. Dormido "*A network of automatic control web-based laboratories*", IEEE Trans. LearningTechnol., vol. 4, no. 3, pp.197 -208 2011.

[9] Hurley, W.G., y "*Development, implementation, and assessment of a web-based power electronics laboratory*," Education, IEEE Transactions on , vol.48, no.4, pp.567,573, Nov. 2005 Chi Kwan Lee.

[10] Sivakumar, S.C., W. Robertson, M. Artimy, y N., "*A web-based remote interactive laboratory for Internetworking education*," Education, IEEE Transactions on , vol.48, no.4, pp.586,598, Nov. 2005 Aslam.

[11] Spanias, A., y V., "*Interactive online undergraduate laboratories using J-DSP*," Education, IEEE Transactions on , vol.48, no.4, pp.735,749, Nov. 2005 Atti.

[12] Torres, F, F.A. Candelas, S.T. Puente, J. Pomares, P. Gil, y F.. "*Experiences with Virtual Environment and Remote Laboratory for Teaching and Learning Robotics at the University of Alicante*". International Journal of Engineering Education (ISSN 09 Ortíz.

[13] *XAMPP*. https://www.apachefriends.org/es/index.html (último acceso: 6 de Junio de 2015).

[14] *Apache*. http://apache.org/ (último acceso: 24 de Junio de 2015).

[15] *PHP*. http://www.php.net/ (último acceso: 24 de Junio de 2015).

[16] *NetBeans IDE*. https://netbeans.org/ (último acceso: 24 de Junio de 2015).

[17] MySQL. *Chapter 14 The InnoDB Storage Engine.* http://dev.mysql.com/doc/refman/5.6/en/innodb-storage-engine.html (último acceso: 16 de Junio de 2015).

[18] *phpMyAdmin.* http://www.phpmyadmin.net (último acceso: 21 de Mayo de 2015).

[19] *Xilinx ISE Design Suite.* http://www.xilinx.com/products/design-tools/ise-design-suite.html (último acceso: 19 de Junio de 2015).

[20] *ModelSim.* http://www.mentor.com/products/fv/modelsim/ (último acceso: 12 de Junio de 2015).

[21] *PicoBlaze 8-bit Embedded Microcontroller.* http://www.xilinx.com/support/documentation/ip_documentation/ug129.pdf (último acceso: 29 de Mayo de 2015).

[22] Osciloscopio MSOX3014A. http://www.keysight.com/en/pd-1947946-pn-MSOX3014A/oscilloscope-100-mhz-4-analog-plus-16-digital-channels?nid=-33573.970760&cc=ES&lc=eng (último acceso: 25 de Mayo de 2015).

[23] *Hash de contraseñas PHP*. http://php.net/manual/es/faq.passwords.php (último acceso: 21 de Mayo de 2015).

ANEXO I: CONFIGURACIÓN DEL SERVIDOR

En este anexo se detallan las configuraciones que se han aplicado en uno de los ordenadores que actuará de servidor para el despliegue del laboratorio remoto, así como las especificaciones que los administradores deben tener en cuenta.

I.1 Servidor web Apache

Una vez instalado el paquete XAMPP, el servidor web de Apache tiene un archivo principal de configuración llamado *httdp.conf*. En él pueden modificarse parámetros globales y directivas de funcionamiento.

Para el laboratorio remoto desarrollado se ha creado una carpeta en un directorio del servidor para almacenar las páginas web de gestión de la base de datos y de la interfaz web, separándolas en directorios diferentes.

Como se explicó anteriormente, la web de gestión de la base de datos solo podrá ser accedida desde el puesto servidor por un administrador. El archivo *httpd.conf* permite asignar diferentes permisos a los directorios que se especifiquen.

Por defecto, Apache viene configurado denegando el acceso a todo el sistema de archivos del servidor como muestra la siguiente especificación:

```
<Directory />
    AllowOverride none
    Require all denied
</Directory>
```

La primera directiva que modificamos es `DocumentRoot`, que especifica el directorio desde donde queremos servir los documentos, en este caso por ejemplo, alojaremos todas las páginas necesarias en una carpeta llamada Servidor.

```
DocumentRoot "C:/Servidor"
```

A continuación se dan permisos a las subcarpetas del directorio Servidor. La primera carpeta llamada "**insertarBD**" contiene las páginas para la gestión de la base de datos, que sólo podrá ser accedida desde dicho servidor y por nadie ajeno.

```
<Directory "C:/Servidor/insertarBD">
    Options Indexes FollowSymLinks Includes ExecCGI
    AllowOverride None
    Require local
</Directory>
```

Las opciones y directivas más importantes que se detallan para este directorio en concreto son:

- *Indexes*: permite mostrar la lista del contenido de este directorio en caso de que la página de inicio (en este caso, inicioBD.php) no se encuentre.
- *FollowSymLinks*: permite seguir enlaces simbólicos para este directorio.
- *Includes*: permite añadir contenido generado dinámicamente a las páginas HTML a servir.
- *AllowOverride None*: evitamos que por cada petición que se haga al servidor se busque y lea en el directorio otro posible archivo de configuración específico para él (llamado *.htaccess*), mejorando así el rendimiento del servidor.
- *Require local*: solo permite acceder localmente donde esté instalado el servidor.

Para la carpeta "**labRemoto**" que contiene las páginas de la interfaz web, se han asignado estos permisos:

```
<Directory "C:/Servidor/labRemoto">
    Options Includes ExecCGI
    AllowOverride None
    Require all granted
</Directory>
```

La directive *Require all granted* permite el acceso a cualquier usuario, puesto que deberá ser accesible por todos los alumnos independientemente de su localización.

Puesto que las páginas de inicio de cada uno de los directorios tienen nombres distintos a los tradicionales (insertarBD.php e inicio.php), también se ha modificado la directiva *DirectoryIndex*. Esta directiva hace que el servidor muestre alguna página que se especifique dentro de ella cuando se accede a un directorio. En este caso las añadimos a todas las predeterminadas:

```
<IfModule dir_module>
    DirectoryIndex index.php index.pl index.cgi index.asp index.shtml
index.html index.htm \
                   default.php default.pl default.cgi default.asp
default.shtml default.html default.htm \
                   home.php home.pl home.cgi home.asp home.shtml home.html
home.htm inicio.php inicioBD.php
</IfModule>
```

Por último, para la puesta final del laboratorio remoto, deberían modificarse los parámetros de `ServerName` especificando el nombre y puerto que el servidor utiliza para identificarse y `ServerAdmin` con la dirección de correo a la que se deben enviar los problemas de uso del servidor.

I.2 php.ini

Es el fichero de configuración que se lee al arrancar PHP. Se han modificado algunos valores de los que están predeterminados inicialmente:

- `max_execution_time`: establece el tiempo máximo que se puede ejecutar la página antes de interrumpirse. Puesto que la generación del archivo .bit y programación de la FPGA puede demorar algunos minutos, su valor se ha establecido en 240 segundos.
- `max_input_time`: es el tiempo máximo que se permite a un script para analizar datos de entrada (POST o GET). Se ha establecido en 60 segundos.
- `session.use_only_cookies`: solo se usarán *cookies* para almacenar el id de sesión en la parte del cliente, evitando ataques que implican pasar el id de sesión en la URL. Será activado a '1'.
- `session.cookie_httponly`: marca la *cookie* como accesible sólo a través del protocolo HTTP y no por lenguajes de script, reduciendo la posibilidad de robos de identidad a través de ataques.
- `post_max_size`: '2M'. Máximo tamaño de los datos de POST permitido a 2Mbytes, tamaño suficiente para los archivos que se esperan del usuario.
- `display_startup_errors`, `track_errors`: Off. Es aconsejable desactivarlo en entornos de producción.
- `session.bug_compat_42`, `session.bug_compat_warn`: Off. Antiguas versiones de PHP tienen un error que permite inicializar una variable de sesión en el ámbito global aunque register_globals esté desactivado. Se recomienda desactivar el *warning* que generaría.

Otros valores convenientes de comprobar su valor para el correcto funcionamiento del laboratorio son:

- `file_uploads`: On. Permite la subida de ficheros mediante HTTP.
- `max_file_uploads`: '20'. Es el número de archivos que permite subir al usuario simultáneamente. El valor es suficientemente holgado para los archivos que subiría un alumno a esta interfaz web.
- `upload_max_file_size`: '2M'. Permite subir archivos de hasta 2Mbytes.
- `display_Errors`: Off. Evitamos mostrar los errores al usuario.

- `register_globals`: Off. Evita que se generen variables globales para *cookies* y valores enviados por GET y POST que podrían ser accedidas mediante la URL malintencionadamente.
- `short_open_tag`: para evitar el uso abreviado de las etiquetas de apertura de PHP (<? ?>) debe ser '0'. Podría generar conflictos con el uso de XML.

I.3 Configuración de variables del laboratorio remoto

Como se ha ido explicando a lo largo de este TFG, las pruebas se han realizado con unos valores de tiempo y directorios específicos que podrán ser cambiados por el administrador cuando desee otro funcionamiento.

En primer lugar, al inicio de la página **funciones.php** (ver Anexo II.2.15) funciones.php se pueden cambiar los valores temporales para la interfaz, la ip del osciloscopio y en caso de que se establezca otro puerto serie también deberá ser modificado.

```
<?php
// Definicion de variables globales

// Tiempo máximo de uso de la interfaz para el alumno (en segundos)
global $t_interfaz;
       $t_interfaz = 1200; // 20 min
// Tiempo máximo de permanencia de la reserva en la BD (en segundos)
global $t_reserva;
       $t_reserva = 7200;  // 2 horas
// Número de segundos en un día
global $t_dia;
       $t_dia = 86400;
// Asignación para la comprobación de permanencia en la reserva teniendo
en cuenta
// el cambio de día
global $t_reserva_fin_dia;
       $t_reserva_fin_dia = $t_dia - $t_reserva;

// Asignación de la ip del osciloscopio
global $ip_osciloscopio;
       $ip_osciloscopio = "http://xxx.xxx.xxx.xxx";

// Directorio donde se encuentran alojadas las páginas php
global $pathInclude;
       $pathInclude =(string)"/labRemoto/";
// Puerto COM donde se instala el FTDI para la conversión USB-UART
global $puertoCOM;
       $puertoCOM = 'COM8:';
```

También deberán cambiarse las constantes para la base de datos:

```
// Definición de constantes para la BD
// Máximo valor numérico de la contraseña
define("MAX_NUM_PASS", 99999999);
// Alojamiento al que nos conectamos
define("HOST", "localhost");
// Nombre de usuario de la Base de Datos
define("USER", "epsuam");
// Contraseña de la Base de Datos
define("PASSWORD", "xxxxxxxx");
// Nombre de la Base de Datos utilizada
define("DATABASE", "hctlab");
```

En la función **asignaciónRutas( )** dentro de la misma página **funciones.php** se deben establecer correctamente las rutas de programas utilizados y ficheros que son necesarios adjuntar a los que el alumno sube al servidor como se explicó en el apartado 2.

```
function asignacionRutas(){

    // Creamos un array para listar todas las rutas predefinidas
    $path = [];

    if(isset($_SESSION['usuario'])){
        // 1) Ruta para la creación del proyecto de cada alumno
        $path[] = "C:/Servidor/labRemoto/". $ SESSION['usuario'];
        // 2) Ruta donde se guardarán los ficheros que nos envía el alumno
        $path[] = "C:/Servidor/labRemoto/". $_SESSION['usuario'] .
"/files/";
        // 3) Ruta de la librería ieee_proposed remota para compilar
        $path[] = 'C:\Prueba_TFG\ieee_proposed\\';
        // 4) Ruta donde almacenamos los ficheros de nuestro servidor
necesarios para la compilación y programación de la FPGA
        $path[] = "C:/Prueba_TFG/";
        // 5) Ruta donde está instalado Xilinx
        $path[] = 'C:\Xilinx\14.4\ISE_DS\ISE\bin\nt64';
        // 6) Ruta donde está el ejecutable para leer el puerto
        $path[] = 'C:\Prueba_TFG\UART_2.exe';

        return $path;
    }else{
        return 0;
    }
}
```

Además, si se establecen otros directorios para la actual carpeta de **labRemoto**, deberán actualizarse las rutas de las cabeceras de las páginas de los Anexos II.2.2) cabecera.inc, II.2.13) cabeceraLab.inc y II.2.14) cabeceraLab_1.inc.

Por último, si el valor del puerto serie cambia del actual COM8, deberá modificarse la siguiente línea del archivo '**UART_2.cpp'** del Anexo II.2.18) UART_2.cpp:

```
TCHAR *pcCommPort = TEXT("COM8");
```

Así como si se establecen otras configuraciones del puerto serie, deben modificarse las siguientes líneas con los nuevos parámetros del puerto:

```
    dcb.BaudRate = CBR_19200;      //  baud rate
    dcb.ByteSize = 8;              //  data bits
    dcb.Parity   = NOPARITY;       //  parity bit
    dcb.StopBits = ONESTOPBIT;     //  stop bit
```

ANEXO II: LISTA DE CÓDIGOS

En este anexo se lista y muestra el código implementado para este TFG. Como se presentó en el apartado 2 del ESQUEMA GENERAL, el servidor necesita alojar diferentes archivos para el funcionamiento del laboratorio remoto.

El código se dividirá según los módulos que se han presentado a lo largo de esta memoria.

Módulo de comunicación UART:

- **top_1.vhd**: es el top level de la comunicación USB-UART que permitirá recibir los 3 bytes desde la interfaz web y transmitir el valor de los LEDs. Instancia los módulos con la funcionalidad de la UART, así como el Controlador, que será provisto por los alumnos desde la web.
- **Top.ucf:** fichero con las asignaciones de los pines de entrada y salida de la FPGA.

Desarrollo en servidor web:

- **Carpeta labRemoto**: contiene las páginas .php necesarias para la ejecución del laboratorio remoto y la aplicación en C++ para lectura del puerto. Se detallaron una a una en el apartado de resultados.
- **Carpeta insertarBD**: contiene las páginas .php para la gestión y actualización de la base de datos para los administradores.

II.1 Módulo de comunicación UART

II.1.1) top_1.vhd

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;
use IEEE.std_logic_arith.all;
use IEEE.math_real.all;

entity top_1 is
      port(
      Clk: in std_logic; --Reloj del sistema
      Reset: in std_logic; --Señal de Reset
      Serial_in: in std_logic; --Dato de entrada desde el PC
      Serial_out: out std_logic; --Dato de salida hacia el PC
      ADCIn : in  STD_LOGIC_VECTOR (7 downto 0); --Entrada del ADC
      PWM : out  STD_LOGIC; --Señal de PWM
      nCS : out  std_logic; --Control para el ADC
      nRD : out  std_logic; --Control para el ADC
      nCONV : out  std_logic; --Control para el ADC
      LEDs : out  STD_LOGIC_VECTOR (7 downto 0)); --Salida a LEDs para
depuracion

end top_1;

architecture Behavioral of top_1 is

component uart_tx
            port (
                              data_in : in std_logic_vector(7 downto 0);
                              write_buffer : in std_logic;
                              reset_buffer : in std_logic;
                              en_16_x_baud : in std_logic;
                                serial_out : out std_logic;
                               buffer_full : out std_logic;
                           buffer_half_full : out std_logic;
                                       Clk : in std_logic);
end component uart_tx;

component uart_rx
            port (
                             serial_in : in std_logic;
                              data_out : out std_logic_vector(7 downto 0);
                           read_buffer : in std_logic;
                          reset_buffer : in std_logic;
                          en_16_x_baud : in std_logic;
                    buffer_data_present : out std_logic;
                           buffer_full : out std_logic;
                      buffer_half_full : out std_logic;
                                   Clk : in std_logic);
end component uart_rx;
```

```
component Controlador is
    Port (
                    Clk : in  STD_LOGIC; --Reloj del sistema
                    Reset : in  STD_LOGIC; --Señal de Reset
                    Swap : in  STD_LOGIC; --Selector de valor para PWM. '0'
activa el lazo cerrado. '1' lazo abierto.
                    ADCIn : in  STD_LOGIC_VECTOR (7 downto 0); --Entrada del
ADC
                    LEDs : out  STD_LOGIC_VECTOR (7 downto 0); --Salida a
LEDs para depuracion
                    Consigna : in  STD_LOGIC_VECTOR (7 downto 0); --Consigna
del lazo
                    CapturaConsigna : in  STD_LOGIC; --Captura de Consigna
                    PWM : out  STD_LOGIC; --Señal de PWM
                    nCS : out  std_logic; --Control para el ADC
                    nRD : out  std_logic; --Control para el ADC
                    nCONV : out  std_logic); --Control para el ADC
end component Controlador;

-- Declaración de señales
signal Consigna: std_logic_vector(7 downto 0);
signal Swap : std_logic;
signal CapturaConsigna : std_logic := '1';
signal Reset_regulador : std_logic;

signal baud_count : integer range 0 to 163;
signal enable_baud : std_logic;

constant MAX_TX_COUNT : integer := 1000000; -- Cada 0.02s
-- Señal para controlar cada cuánto tiempo se envía un dato hacia la
interfaz web
signal write_count  : integer range 0 to MAX_TX_COUNT; -- (f = 50MHz -> T
= 20ns)

------------------------ Señales TX ------------------------
signal write_buffer_tx: std_logic;
signal data_in_tx: std_logic_vector(7 downto 0);

------------------------ Señales RX ------------------------

signal uart_rx_bytes_received: integer := 0;
signal trama_completa : std_logic := '0';
signal data_out_rx : std_logic_vector(7 downto 0);
signal buffer_data_present_rx : std_logic;
signal read_buffer_rx: std_logic := '1';
signal ResetUART: std_logic;

-- Creamos un array para recibir los 3 bytes de datos desde la web a la
FPGA
type RX_ARRAY is array(2 downto 0) of std_logic_vector(7 downto 0);
signal web_rx_frame: RX_ARRAY;

-- Señal para poder descartar tramas que no lleguen conjuntas al array
signal contadorTimeout : integer;
constant MAX_TIMEOUT : integer := 50000;

begin

Control : Controlador
port map (
                    Clk => Clk,
```

```
                Reset => Reset_regulador,
                 Swap => Swap,
                ADCIn => ADCIn,
                 LEDs => LEDs,
             Consigna => Consigna,
      CapturaConsigna => '1', -- Siempre estaremos capturando la consigna
                  PWM => PWM,
                  nCS => nCS,
                  nRD => nRD,
                nCONV => nCONV
       );


transmisor: uart_tx
port map (
              data_in => data_in_tx,
         write_buffer => write_buffer_tx,
         reset_buffer => Reset,
         en_16_x_baud => enable_baud,
           serial_out => Serial_out,
          buffer_full => open,
     buffer_half_full => open,
                  Clk => Clk);

receptor: uart_rx
port map (
            serial_in => Serial_in,
             data_out => data_out_rx, --Data Out debera gestionarse de
manera que recibo 4 bytes desde la Web:
                                                                        --
1) Swap
                                                                        --
2) Consigna (in 7:0) es el valor de los interruptores
                                                                        --
3) Reset (hacer reset general con or Reset de botón FPGA)
          read_buffer => read_buffer_rx,
         reset_buffer => Reset,
         en_16_x_baud => enable_baud,
   buffer_data_present => buffer_data_present_rx,
          buffer_full => open,
     buffer_half_full => open,
                  Clk => Clk);

-- Asignación para hacer Reset del Controlador

Reset_regulador <= Reset or ResetUART;

-- Proceso para contar el número de bytes que he recibido y poder activar
-- el procesamiento de la trama
numeroBytesRecibidos : process(Clk,Reset)
begin
      if Reset = '1' then
            uart_rx_bytes_received <= 0;

      elsif (clk'event and Clk='1') then
            if contadorTimeout = MAX_TIMEOUT then
                  uart_rx_bytes_received <= 0;
            elsif uart_rx_bytes_received = 3 then
                  uart_rx_bytes_received <= 0;
            else
```

```
                  if buffer_data_present_rx = '1' then
                   uart_rx_bytes_received <= uart_rx_bytes_received + 1;
                  end if;
            end if;
      end if;
end process numeroBytesRecibidos;

-- Proceso para asignar los datos recibidos de la Web a la FPGA y
gestionar ResetUART
process(Clk,Reset)
begin
      if Reset = '1' then
            Swap <= '0';
            Consigna <= (others => '0');
            ResetUART <= '1';
      elsif (clk'event and Clk='1') then
            if trama_completa = '1' then
                  Swap        <= web_rx_frame(0)(0); --  Asigno el bit
menos significativo que recibo
                  Consigna <= web_rx_frame(1); -- Se le asignan los bits
que he enviado desde la web directamente
                  if web_rx_frame(2) = x"01" then
                        ResetUART <= '1';
                  else
                        ResetUART <= '0';
                  end if;
            else
                  ResetUART <= '0';
            end if;
      end if;
end process;

-- Proceso con contador para que si en 1ms no se reciben datos conjuntos
-- se descarten esas tramas, pues esperamos 3 bytes seguidos

descarteTramas : process(Clk, Reset)
      begin
            if Reset = '1' then
                  contadorTimeout <= 0;
            elsif (clk'event and Clk='1') then
                  if buffer_data_present_rx = '1' then
                        contadorTimeout <= 0;
                  else
                        contadorTimeout <= contadorTimeout + 1;
                  end if;
            end if;
end process descarteTramas;

-- Proceso de gestión de los datos recibidos desde el puerto serie hacia
la FPGA
recepcionDatos: process(Clk, Reset)
      begin
            if Reset = '1' then
                  trama_completa <= '0';
                  for i in 0 to 2 loop
                        web_rx_frame(i) <= (others => '0');
                  end loop;
            elsif (clk'event and Clk='1') then
                  -- Si hay dato recibido en el buffer
                  if buffer_data_present_rx = '1' then
                        -- Lo asignamos a una posición del array de datos
```

```
                        web_rx_frame(uart_rx_bytes_received)<=data_out_rx;
                        read_buffer_rx <= '1';
                              -- Hemos recibido todos los bytes
                        if uart_rx_bytes_received = 2 then
                              -- Enviamos la trama
                              trama_completa <= '1';
                        else
                              trama_completa <= '0';
                        end if;
                  else
                        trama_completa <= '0';
                  end if;
            end if;
end process recepcionDatos;

-- Proceso con contador para obtener la frecuencia adecuada de
en_16_x_baud de la UART

Baud_divisor : process(Clk, Reset)
begin
      if Reset = '1' then
            baud_count <= 0;
            enable_baud <= '0';
      elsif (clk'event and Clk='1') then
            if baud_count = 162 then
          baud_count <= 0;
          enable_baud <= '1';
      else
          baud_count <= baud_count + 1;
          enable_baud <= '0';
            end if;
   end if;
end process Baud_divisor;

-- Proceso para la transmisión del dato hacia la interfaz Web
Envio_dato : process(Clk, Reset)
begin
      if Reset = '1' then
            data_in_tx <= x"00";
   elsif (clk'event and Clk='1') then
            if write_buffer_tx = '1' then
                  data_in_tx <= ADCin;
                  end if;
            end if;
end process Envio_dato;

-- Proceso para trasmitir el dato de forma que no se llene el buffer
-- Se utiliza un contador de valor elevado que active un solo pulso
-- la señal write_buffer

Buffer_contador : process(Clk, Reset)
begin
      if Reset = '1' then
                  write_count  <= 0;
                  write_buffer_tx <= '0';
      elsif(clk'event and Clk='1') then

            if(write_count < MAX_TX_COUNT) then
                  write_count  <= write_count + 1;
                  write_buffer_tx <= '0';
```

```
            else
                  write_buffer_tx <= '1';
                  write_count <= 0;
            end if;
      end if;
end process Buffer_contador;

end Behavioral;
```

II.1.2) top.ucf

```
#Created by Constraints Editor (xc3s1000-ft256-4) - 2014/11/14
NET "Clk" TNM_NET = Clk;
TIMESPEC TS_Clk = PERIOD "Clk" 20 ns HIGH 50%;

NET "Clk" LOC = "T9";
NET "Reset" LOC = "L14";

#Conector A2: Pines para el ADC
NET "nCONV" LOC = "A7";
NET "nCS" LOC = "A8";
NET "nRD" LOC = "B10";
NET "PWM" LOC = "B11";

NET "ADCIn[0]" LOC = "D5";
NET "ADCIn[1]" LOC = "D6";
NET "ADCIn[2]" LOC = "E7";
NET "ADCIn[3]" LOC = "D7";
NET "ADCIn[4]" LOC = "D8";
NET "ADCIn[5]" LOC = "D10";
NET "ADCIn[6]" LOC = "B4";
NET "ADCIn[7]" LOC = "B5";

#Conector B1: Pines para la UART
NET "Serial_in" LOC = "T3"; #Conector B1, pin 5
NET "Serial_out" LOC = "P10";#Conector B1, pin 9

#Pines de los LED
NET "LEDs[0]" LOC = "K12";
NET "LEDs[1]" LOC = "P14";
NET "LEDs[2]" LOC = "L12";
NET "LEDs[3]" LOC = "N14";
NET "LEDs[4]" LOC = "P13";
NET "LEDs[5]" LOC = "N12";
NET "LEDs[6]" LOC = "P12";
NET "LEDs[7]" LOC = "P11";
```

II.2 Desarrollo en servidor web: labRemoto

II.2.1) inicio.php

```
<!-- inicio.php -->
<?php include("/include/cabecera.inc"); ?>

<div class="cont-ppal">
    <div class="cont">
        <h1 class="cont-cabecera"> Laboratorio <br/> Sistemas <br/> de
<br/> Control</h1>
    </div>
</div>
<div class="content-wrapper">
    <div class="content">

<?php include("/include/pie.inc");?>
```

II.2.2) cabecera.inc

```
<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <link rel="stylesheet" href="http://netdna.bootstrapcdn.com/font-
awesome/4.0.3/css/font-awesome.css">
    <link rel="stylesheet"
href="http://yui.yahooapis.com/pure/0.6.0/grids-responsive-min.css">
    <link rel="stylesheet" href="http://yui.yahooapis.com/pure/0.6.0/pure-
min.css">
    <link rel="stylesheet" href="/labRemoto/include/style.css">

    <title>Laboratorio Sistemas de Control</title>

</head>
<body>
    <div class="header">
        <div class="home-menu pure-menu pure-menu-horizontal pure-menu-
fixed">
            <a class="pure-menu-heading" href="inicio.php">Inicio</a>
            <a class="pure-menu-heading" href="calendario.php">Acceso a
calendario</a>
            <a class="pure-menu-heading" href="borrado.php">Borrar
reserva</a>
            <a class="pure-menu-heading" href="validacion.php">Acceso a
interfaz</a>
        </div>
    </div>
```

II.2.3) pie.inc

```
    <div class="footer l-box is-center">
        Human Computer Technology Laboratory | Escuela Politécnica
Superior | Universidad Autónoma de Madrid
    </div>
</div>
</div>

</body>
</html>
```

II.2.4) calendario.php

```
<!-- calendario.php -->

<?php   include("/include/cabecera.inc");
        include("/include/funciones.php");?>

<div class="cont-ppal">
    <div class="cont-calendar">
        <div class="cont-cabecera-calendar">
            <div class="pure-form ">
                <form action ="reserva.php" method = "post" enctype =
"multipart/form-data">
                    <div class="input-group margin-bottom-sm">
                    <span class="input-group-addon"><i class="fa fa-
envelope-o fa-fw"></i></span>
                    <input class="form-control" type="text" name =
"emailreserva" placeholder="Usuario" size="30">
                    </div>
                <br />
                    <div class="input-group">
                    <span class="input-group-addon"><i class="fa fa-key
fa-fw"></i></span>
                    <input class="form-control" type="password" name =
"passreserva" placeholder="Contraseña">
                    </div>
                <br />
                <button type ="submit" class="button-success button-
success-green pure-button"><b>Entrar</b></button>
                </form>
            `</div>
        </div>
    </div>
</div>
 <div class="content-wrapper">
    <div class="content">

<?php include("/include/pie.inc");?>
```

II.2.5) reserva.php

```
<!-- reserva.php -->
<?php include("/include/cabecera.inc");
      include("/include/funciones.php");?>

<div class="cont-ppal">
    <div class="cont-reserva">
        <h1 class="cont-cabecera-reserva">

<?php

    // Validamos los datos recibidos del formulario
    $usuario_calendario = filter_input(INPUT_POST, 'emailreserva',
FILTER_VALIDATE_EMAIL);
    $pass_calendario    = filter_input(INPUT_POST, 'passreserva',
FILTER_VALIDATE_INT,
                           ['min_range' => 1, 'max_range' =>
MAX_NUM_PASS]);

    // Comprobación de que es un email válido y perteneciente a la UAM
    if(isset($usuario_calendario) && isset($pass_calendario) &&
            preg_match('/(@estudiante.uam.es)/', $usuario_calendario)){
        //Conectamos a la BBDD para comprobar si el usuario tiene acceso
        $conexion = conectarDB();
        $sentencia = "SELECT hash FROM usuarios WHERE `email`=?";
        if($stmt = mysqli_prepare($conexion, $sentencia)){
            // Agregamos la variable $usuario_calendario como parámetro a
la sentencia preparada
            mysqli_stmt_bind_param($stmt, 's', $usuario_calendario);
            if(!mysqli_stmt_execute($stmt)){
                reservaErrorUsuarioBD($conexion, $pathInclude);
            }else{
                // Almacenamos el resultado
                mysqli_stmt_store_result($stmt);
                // Vinculamos el resultado a una variable
                mysqli_stmt_bind_result($stmt, $hashedPassBD);
                // Obtenemos el resultado(hash de la contraseña de la Base
de Datos
                mysqli_stmt_fetch($stmt);
                // Si se verifica la contraseña introducida por el usuario
con su hash, proseguimos
                if(password_verify($pass_calendario, $hashedPassBD)){
                    ?>
                    Seleccione la hora que desea reservar para acceder a
la interfaz Web del laboratorio </h1>
                    <div class="pure-form">
                    <form action ="reservaOk.php" method = "post">
                    <input type="hidden" name = "email" value = "<?php
echo $usuario_calendario;?>">
                    <select name="hora">
                        <option selected="selected" value="1">1</option>
                        <option value="2">2</option>
                        <option value="3">3</option>
                        <option value="4">4</option>
                        <option value="5">5</option>
                        <option value="6">6</option>
                        <option value="7">7</option>
                        <option value="8">8</option>
                        <option value="9">9</option>
```

```
                        <option value="10">10</option>
                        <option value="11">11</option>
                        <option value="12">12</option>
                        <option value="13">13</option>
                        <option value="14">14</option>
                        <option value="15">15</option>
                        <option value="16">16</option>
                        <option value="17">17</option>
                        <option value="18">18</option>
                        <option value="19">19</option>
                        <option value="20">20</option>
                        <option value="21">21</option>
                        <option value="22">22</option>
                        <option value="23">23</option>
                        <option value="24">24</option>

                    </select>
                    <select name="min">
                        <option selected="selected" value="1">00</option>
                        <option value="20">20</option>
                        <option value="40">40</option>
                    </select><br /><br />
                    <button type="submit" class="button-succes button-
success-green pure-button"><b>Reservar</b></button>
                    </form>
                    </div> <?php

                    unset($hashedPassBD);
                    mysqli_stmt_close($stmt);
                    mysqli_close($conexion);
                }else{
                    unset($hashedPassBD);
                    reservaErrorUsuario($stmt, $conexion, $pathInclude);
                }
            }
        }else{
            reservaErrorUsuarioPass($pathInclude);
        }
    }else{
        reservaErrorUsuarioPass($pathInclude);
    }
?>
    </div>
</div>
<div class="content-wrapper">
    <div class="content">

<?php include("/include/pie.inc");?>
```

II.2.6) reservaOk.php

```
<!-- reservaOk.php -->

<?php include("/include/cabecera.inc");
      include("/include/funciones.php");?>
<div class="cont-ppal">
    <div class="cont-reserva">
        <h1 class="cont-cabecera-reserva">
<?php

    $form = filter_input_array(INPUT_POST,
            ['email' => FILTER_VALIDATE_EMAIL,
             'hora'  => ['filter' => FILTER_VALIDATE_INT,
                         'flags'  => FILTER_REQUIRE_SCALAR,
                         'options'=> ['min_range' => 1, 'max_range' =>
24]],
             'min'   => ['filter' => FILTER_VALIDATE_INT,
                         'flags'  => FILTER_REQUIRE_SCALAR,
                         'options'=> ['min_range' => 1, 'max_range' =>
50]]]);

    if(isset($form) && $form['email'] && $form['hora'] && $form['min']){

        // Para evitar errores de no validación por ser un 0, ahora
asignamos
        // el minuto 00 en caso de que esa opción fuese escogida
        if($form['hora'] == 24){
            $form['hora'] = "00";
        }
        if($form['min'] == 1){
            $form['min'] = "00";
        }

        // Construimos las variables y la hora de reserva en formatos
adecuados para las BBDD
        $email = $form['email'];
        $hora_reserva_ok = $form['hora'] . ":" . $form['min'] . ":00";

        $hora_reserva_s = ($form['hora'] * 3600) + ($form['min'] * 60);
        $hora_actual = date("H:i:s");
        list($h, $m, $s) = explode(':',$hora_actual);
        $hora_actual_s = ($h * 3600) + ($m * 60) + $s;

        if(($hora_actual_s > $hora_reserva_s) && $hora_reserva_s != 0 ){
            reservaOkErrorHoraActual($pathInclude);
        }else{
            //Conectamos a la BBDD para almacenar la hora de reserva para
el usuario
            $conexion = conectarDB();
            // Obtenemos las horas reservadas para el día actual de ese
alumno
            $sentencia = "SELECT hora_reserva, email FROM reserva WHERE
fecha_reserva = CURDATE()";
            if($stmt = mysqli_prepare($conexion, $sentencia)){
                if(!mysqli_stmt_execute($stmt)){
                    reservaOkErrorAlmacenar($conexion, $pathInclude);
                }else{
```

```
                    // Almacenamos el resultado
                    mysqli_stmt_store_result($stmt);

                    $num_rows = mysqli_stmt_num_rows($stmt);
                    if($num_rows != 0){
                        // Si ya había reservas, comprobamos que la nueva
hora de reserva
                        // mantenga el margen con t_reserva
                        // Vinculamos los resultados
                        mysqli_stmt_bind_result($stmt, $row, $row_2);
                        // Obtenemos la lista de email y horas reservadas
en arrays
                        while(mysqli_stmt_fetch($stmt)){
                            $horas_reservadas[] = $row;
                            $email_reservadas[] = $row_2;
                        }

                        $flag = [];
                        for($i = 0; $i<$num_rows; $i++){

                            list($h, $m, $s) =
explode(':',$horas_reservadas[$i]);
                            $horas_reservadas_s[$i] = ($h * 3600) + ($m *
60) + $s;
                            $fecha_actual = date("Y-m-d");

                            // Asignaciones para la hora de 00:00:00
                            if($horas_reservadas_s[$i] == 0){
                                $horas_reservadas_s[$i] = $t_dia;
                            }elseif($hora_reserva_s == 0){
                                $hora_reserva_s = $t_dia;
                            }

                            // Comprobaciones según la hora seleccionada
de reserva
                            // Hora de reserva válida(1): no hay horas
reservadas para ese usuario
                            // y no coinciden con la hora de reserva
escogida ni está entre los
                            // tiempos de espera de reserva
                            if($hora_reserva_s ==
$horas_reservadas_s[$i]){
                                $flag[] = 2;
                            }elseif ($hora_reserva_s <
$horas_reservadas_s[$i]){
                                if(($hora_reserva_s >
($horas_reservadas_s[$i] - $t_reserva)) &&
                                        ($email_reservadas[$i] ==
$email)){
                                    $flag[] = 0;
                                }else{
                                    $flag[] = 1;
                                }
                            }elseif($hora_reserva_s >
$horas_reservadas_s[$i]){
                                if(($hora_reserva_s <
($horas_reservadas_s[$i] + $t_reserva)) &&
                                        ($email_reservadas[$i] ==
$email)){
                                    $flag[] = 0;
                                }else{
```

```
                                    $flag[] = 1;
                                }
                            }else{
                                $flag[] = 1;
                            }
                        }

                        // Comprobamos si hay impedimentos para la reserva
según los valores de la flag
                        if(in_array(2,$flag)){
                            reservaOkErrorHoraReservada($stmt, $conexion,
$pathInclude);
                        }elseif(in_array(0,$flag)){
                            reservaOkErrorEstaReservado($stmt, $conexion,
$pathInclude, $t_reserva);
                        }elseif(in_array(1,$flag)){
                            // Sí se puede reservar
                            $sentencia = "INSERT INTO
reserva(`email`,`fecha_reserva`,`hora_reserva`,`acceso`) VALUES "
                                        . "(?,CURDATE(),?,0)";
                            if($stmt = mysqli_prepare($conexion,
$sentencia)){

                                // Agregamos las variables como parámetros
a la sentencia preparada
                                mysqli_stmt_bind_param($stmt, 'ss',
$email, $hora_reserva_ok);
                                if(!mysqli_stmt_execute($stmt)){
                                    reservaOkErrorFecha($stmt, $conexion,
$pathInclude);
                                }else{
                                    echo "Reserva efectuada con éxito <br
/><br />";
                                    echo '<a class="button-success button-
success-green pure-button" '
                                    . 'href = "' . $pathInclude .
'inicio.php">Página principal<br /></a></h1>';
                                    mysqli_stmt_close($stmt);
                                    mysqli_close($conexion);
                                }
                            }else{
                                reservaOkErrorAlmacenar($conexion,
$pathInclude);
                            }
                        }
                    }else{
                        // Si no hay reservas previamente realizadas para
ese día, la almacenamos
                        $sentencia = "INSERT INTO
reserva(`email`,`fecha_reserva`,`hora_reserva`,`acceso`) VALUES "
                                    . "(?,CURDATE(),?,0)";
                        if($stmt = mysqli_prepare($conexion, $sentencia)){
                            // Agregamos las variables como parámetros a
la sentencia preparada
                            mysqli_stmt_bind_param($stmt, 'ss', $email,
$hora_reserva_ok);
                            if(!mysqli_stmt_execute($stmt)){
                                reservaOkErrorFecha($stmt, $conexion,
$pathInclude);
                            }else{
```

```
                                echo "Reserva efectuada con éxito <br
/><br />";
                                echo '<a class="button-success button-
success-green pure-button" '
                                . 'href = "' . $pathInclude .
'inicio.php">Página principal<br /></a></h1>';
                                mysqli_stmt_close($stmt);
                                mysqli_close($conexion);
                            }
                        }else{
                            mysqli_stmt_close($stmt);
                            reservaOkErrorAlmacenar($conexion,
$pathInclude);
                        }
                    }
                }
            }else{
                mysqli_stmt_close($stmt);
                reservaOkErrorAlmacenar($conexion, $pathInclude);
            }
        }
    }else{
        echo "Error en los datos recibidos<br /><br />";
        echo "Inténtelo de nuevo o contacte con el administrador<br /><br
/>";
        echo '<a class="button-fail button-fail-orange pure-button" href =
"' . $pathInclude . 'inicio.php">Regresar<br /></a>';
    } ?>
    </div>
</div>
<div class="content-wrapper">
    <div class="content">
<?php include("/include/pie.inc"); ?>
```

II.2.7) borrado.php

```
<!-- borrado.php -->

<?php include("/include/cabecera.inc");
      include("/include/funciones.php");?>

<div class="cont-ppal">
    <div class="cont-calendar">
        <div class="cont-cabecera-calendar">
            <div class="pure-form ">
                <form action ="borrado_1.php" method = "post" enctype =
"multipart/form-data">
                    <div class="input-group margin-bottom-sm">
                    <span class="input-group-addon"><i class="fa fa-
envelope-o fa-fw"></i></span>
                    <input class="form-control" type="text" name
="emailborrado" placeholder="Usuario" size="30">
                    </div><br />
                    <div class="input-group">
                    <span class="input-group-addon"><i class="fa fa-key
fa-fw"></i></span>
                    <input class="form-control" type="password" name
="passborrado" placeholder="Contraseña">
                    </div><br />
```

```
                <button type ="submit" class="button-succes button-
success-green pure-button"><b>Entrar</b></button>
                </form>
            `</div>
        </div>
    </div>
</div>
 <div class="content-wrapper">
    <div class="content">
<?php include("/include/pie.inc");?>
```

II.2.8) borrado_1.php

```
<!-- borradoConfirmacion.php  <h1 class="cont-cabecera-calendar"> -->

<?php include("/include/cabecera.inc");
      include("/include/funciones.php");?>

<div class="cont-ppal">
    <div class="cont-calendar">
        <div class="cont-cabecera-calendar"><?php

    // Validamos el correo y contraseña que se han introducido.
    $form = filter_input_array(INPUT_POST,
            ['emailborrado' => FILTER_VALIDATE_EMAIL,
             'passborrado'  => ['filter' => FILTER_VALIDATE_INT,
                        'flags'  => FILTER_REQUIRE_SCALAR,
                        'options'=> ['min_range' => 1, 'max_range' =>
MAX_NUM_PASS]]]);

    if(isset($form['emailborrado']) && isset($form['passborrado']) &&
            preg_match('/(@estudiante.uam.es)/', $form['emailborrado'])){

        $email = $form['emailborrado'];
        $pass  = $form['passborrado'];
        $hora_actual = horaActual();
        // Conectamos a la BD y comprobamos los datos
        $conexion = conectarDB();
        $sentencia = "SELECT hash FROM usuarios WHERE `email`=?";

        if($stmt = mysqli_prepare($conexion, $sentencia)){
            // Agregamos la variable $email como parámetro a la sentencia
preparada
            mysqli_stmt_bind_param($stmt, 's', $email);
            if(!mysqli_stmt_execute($stmt)){
                borradoOkErrorEmailPass($stmt, $pathInclude);
            }else{
                // Almacenamos el resultado
                mysqli_stmt_store_result($stmt);
                // Vinculamos el resultado a una variable
                mysqli_stmt_bind_result($stmt, $hashedPassBD);
                // Obtenemos el resultado(hash de la contraseña de la Base
de Datos
                mysqli_stmt_fetch($stmt);
                // Si se verifica la contraseña introducida por el usuario
con su hash, proseguimos
                if(password_verify($pass, $hashedPassBD)){
                    $num_rows = mysqli_stmt_num_rows($stmt);
```

```
                    if($num_rows != 0){
                        // Cerramos la anterior sentencia para realizar
una nueva consulta
                        mysqli_stmt_close($stmt);
                        $sentencia = "SELECT hora_reserva FROM reserva
WHERE `email`=?  AND "
                                . "fecha_reserva = CURDATE()";
                        if($stmt = mysqli_prepare($conexion, $sentencia)){
                            mysqli_stmt_bind_param($stmt, 's', $email);
                            if(!mysqli_stmt_execute($stmt)){
                                calendarioErrorSelReserva($stmt,
$conexion, $pathInclude);
                            }else{
                                // Almacenamos el resultado
                                mysqli_stmt_store_result($stmt);
                                // Vinculamos el resultado a una variable
                                mysqli_stmt_bind_result($stmt,
$hora_reserva);
                                $num_rows = mysqli_stmt_num_rows($stmt);

                                // Si no hay reservas realizadas,
informamos
                                if($num_rows == 0){
                                    echo "No tiene una reserva realizada.
<br /><br />";
                                    echo '<a class="button-fail button-
fail-orange pure-button" '
                                    . 'href = "' . $pathInclude .
'inicio.php">Regresar<br /></a>';
                                    mysqli_stmt_close($stmt);
                                    mysqli_close($conexion);
                                }else{

                                    // Almacenamos las reservas
                                    $horas_reserva = [];
                                    while(mysqli_stmt_fetch($stmt)){
                                        $horas_reserva[] = $hora_reserva;
                                    }
                                    $hora_actual = horaActual();
                                    $flag = [];
                                    $horas_de_borrado = [];

                                    for($i = 0; $i<$num_rows; $i++){
                                        // Obtenemos la hora de reserva en
segundos

                                        // para hacer comparaciones
                                        list($h, $m, $s) =
explode(':',$horas_reserva[$i]);

                                        $horas_reserva_s[$i] = ($h * 3600)
+ ($m * 60) + $s;

                                        // Asignamos el caso de las
00:00:00 para operar

                                        if($horas_reserva_s[$i] == 0){
                                            $horas reserva s[$i] = $t dia;
                                        }
                                        // Comprobamos los casos para
obtener las horas

                                        // que sí dejamos borrar al
usuario

                                        if($hora_actual >
$horas_reserva_s[$i]){
```

```
                                        $flag[$i] = 0;
                                    }elseif($hora_actual <
($horas_reserva_s[$i] + $t_reserva) &&
                                            $hora actual >
$horas_reserva_s[$i]){
                                        // Controlamos que la reserva
no se borre en el tiempo de espera
                                        // para evitar uso sucesivo de
reservas
                                        $flag[$i] = 0;
                                    }elseif(($hora_actual >
($horas_reserva_s[$i] + $t_reserva)) ||
                                            $hora_actual <
$horas_reserva_s[$i]){
                                        // En este caso sí se puede
borrar
                                        $horas_de_borrado[] =
$horas_reserva[$i];
                                        $flag[$i] = 1;
                                    }
                                }

                                if($num_rows == 1 && $flag[0] == 0){

                                    echo "No ha transcurrido
suficiente tiempo para borrar su reserva <br /><br />"
                                    . "Inténtelo más tarde. <br /> <br
/>";
                                    echo '<a class="button-fail
button-fail-orange pure-button" '
                                    . 'href = "' . $pathInclude .
'inicio.php">Regresar<br /></a>';
                                    mysqli_stmt_close($stmt);
                                    mysqli_close($conexion);

                                }else{?>
                                    <div class= "nueva">
                                    <form action
="borradoConfirmacion.php" method = "post" enctype = "multipart/form-
data"><br />
                                    Seleccione la hora de reserva que
desea borrar<br /><br />

                                    <?php for($i = 0; $i <
count($horas_de_borrado); $i++){ ?>
                                    <input id= "radio-<?php echo
$i;?>" class="radio-lab" name="borrar" type="radio" value = "<?php echo
$i;?>">
                                    <label for= "radio-<?php echo
$i;?>" class="radio-lab-label"><?php echo $horas_de_borrado[$i]
;?></label>
                                    <input type = "hidden" name =
"horas_reserva_borrado[]" value = "<?php echo $horas_de_borrado[$i];?>">
                                    <?php } ?>
                                    <input id= "radio-<?php echo
$i+1;?>" class="radio-lab" name="borrar" type="radio" value = "100">
                                    <label for= "radio-<?php echo
$i+1;?>" class="radio-lab-label">Ninguna</label>
                                    <input type = "hidden" name =
"emailborradoOk" value = "<?php echo $email;?>">
```

```
                                        <input type = "hidden" name =
"passborradoOk" value = "<?php echo $pass;?>"><br /><br />
                                        <button type= "submit"
class="button-succes button-success-green pure-
button"><b>Borrar</b></button>
                                        </form><?php
                                    }
                                }
                            }
                        }else{
                            echo mysqli_error($stmt);
                            mysqli_stmt_close($stmt);
                        }
                    }else{
                        echo "No tiene reservas realizadas. <br /><br />";
                        echo '<a class="button-fail button-fail-orange
pure-button" '
                        . 'href = "' . $pathInclude .
'borrado.php">Regresar<br /></a>';
                        mysqli_stmt_close($stmt);
                        mysqli_close($conexion);
                    }
                }else{
                    echo "No se ha introducido una contraseña o no es
válida. <br /><br />";
                    echo '<a class="button-fail button-fail-orange pure-
button" '
                    . 'href = "' . $pathInclude .
'borrado.php">Regresar<br /></a>';
                    mysqli_stmt_close($stmt);
                }
            }
        }else{
            borradoOkErrorDatos($pathInclude);
        }
    }else{
        borradoOkErrorDatos($pathInclude);
    }
        ?>
        </div>
    </div>
</div>
</div>
<div class="content-wrapper">
    <div class="content">
<?php include("/include/pie.inc");?>
```

II.2.9) borradoConfirmacion.php

```
<!-- borradoOk.php -->

<?php include("/include/cabecera.inc");
      include("/include/funciones.php");?>

<div class="cont-ppal">
    <div class="cont-calendar">
        <div class="cont-cabecera-calendar"><?php

    // Validamos los datos recibidos del formulario
    $email = filter_input(INPUT_POST, 'emailborradoOk',
FILTER_VALIDATE_EMAIL);
    $pass  = filter_input(INPUT_POST, 'passborradoOk',
FILTER_VALIDATE_INT,
                              ['min_range' => 1, 'max_range' =>
MAX_NUM_PASS]);

    if(isset($email) && isset($pass) &&
preg_match('/(@estudiante.uam.es)/', $email)){

        $indice_borrar = $_POST['borrar'];
        $horas_reserva_borrado = $_POST['horas_reserva_borrado'];
        $hora_actual = horaActual();

        // Si se asignó el valor 100 es que el usuario eligió no borrar
reservas
        if($indice_borrar == 100){
            echo "Ha decidido no borrar reservas.<br /><br />";
            echo '<a class="button-success button-success-green pure-
button" '
                 . 'href = "' . $pathInclude . 'inicio.php">Página
principal</a></h1>';
        }else{

            // Asignamos la hora que se ha seleccionado borrar
            $hora_borrada = $horas_reserva_borrado[$indice_borrar];
            list($h, $m, $s) = explode(':',$hora_borrada);
            $hora_borrada_s = ($h * 3600) + ($m * 60) + $s;

            if($hora_borrada_s == 0){
                $hora_borrada_s = $t_dia;
            }
            // Si la hora actual sigue siendo menor que la que se desea
borrar, accedemos
            if($hora_actual < $hora_borrada_s){

                $conexion = conectarDB();
                // Borramos la reserva de la hora que ha seleccionado
                $sentencia = "DELETE FROM reserva WHERE email=? AND
hora_reserva=? AND "
                        . "fecha_reserva = CURDATE()";
                if($stmt = mysqli_prepare($conexion, $sentencia)){
                    mysqli_stmt_bind_param($stmt, 'ss', $email,
$hora_borrada);
                    if(!mysqli_stmt_execute($stmt)){
                        borradoOkErrorBorrado($stmt);
                    }else{
```

```
                        echo "Se ha borrado correctamente su reserva.<br
/><br />";
                        echo '<a class="button-success button-success-
green pure-button" '
                            . 'href = "' . $pathInclude .
'inicio.php">Página principal</a></h1>';
                        mysqli_stmt_close($stmt);
                        mysqli_close($conexion);
                    }
                }else{
                    borradoOkErrorBorrado($stmt, $pathInclude);
                }
            }else{
                echo "No se permite borrar si la hora actual supera "
                . "la hora que desea borrar.<br /> <br />";
                echo '<a class="button-success button-success-green pure-
button" '
                    . 'href = "' . $pathInclude . 'inicio.php">Página
principal<br /></a></h1>';
            }
        }
    }else{
        borradoOkErrorDatos($pathInclude);
    }
?>
        </div>
    </div>
</div>
<div class="content-wrapper">
    <div class="content">
<?php include("/include/pie.inc");?>
```

II.2.10) validacion.php

```
<!-- validacion.php-->
<?php include("/include/cabecera.inc");
      include("/include/funciones.php");?>

<div class="cont-ppal">
    <div class="cont-calendar">
        <div class="cont-cabecera-calendar">
            <div class="pure-form ">
                <form action ="upload.php" method = "post" enctype =
"multipart/form-data">
                    <div class="input-group margin-bottom-sm">
                    <span class="input-group-addon"><i class="fa fa-
envelope-o fa-fw"></i></span>
                    <input class="form-control" type="text" name
="emailvalidacion" placeholder="Usuario" size="30">
                    </div><br />
                    <div class="input-group">
                    <span class="input-group-addon"><i class="fa fa-key
fa-fw"></i></span>
                    <input class="form-control" type="password" name
="passvalidacion" placeholder="Contraseña">
                    </div><br />
                    <button type ="submit" class="button-success button-
success-green pure-button"><b>Entrar</b></button>
                </form>
```

```
            `</div>
        </div>
    </div>
</div>
 <div class="content-wrapper">
    <div class="content">
<?php include("/include/pie.inc");?>
```

II.2.11) upload.php

```
<!-- upload.php -->
<?php session_start();
      include("/include/cabeceraLab.inc");
      include("/include/funciones.php");?>

<div class="cont-ppal">
    <div class="cont-upload">
        <div class="cont-cabecera-upload">
<?php

    $form = filter_input_array(INPUT_POST,
            ['emailvalidacion' => FILTER_VALIDATE_EMAIL,
             'passvalidacion'  => ['filter' => FILTER_VALIDATE_INT,
                        'flags'  => FILTER_REQUIRE_SCALAR,
                        'options'=> ['min_range' => 1, 'max_range' =>
MAX_NUM_PASS]]]);

    // Variable para controlar el acceso si los datos son bien
introducidos
    $flag = 0;
    // Dependiendo de si acabamos de iniciar sesión o ya estábamos dentro
pero necesitamos
    // subir nuevos ficheros, comprobamos los datos según el caso.
    $hora_actual = horaActual();
    if(isset($_SESSION['estado']) && ($_SESSION['estado'] == 2) &&
            (($_SESSION['hora_reserva'] + $t_interfaz) >  $hora_actual)){

        $email = $_SESSION['usuario'];
        $pass  = $_SESSION['pass'];
        $flag = 1;

    }elseif(isset($form['emailvalidacion']) &&
isset($form['passvalidacion']) && ($form['passvalidacion']!= false) &&
            preg_match('/(@estudiante.uam.es)/',
$form['emailvalidacion'])){
        $email = $form['emailvalidacion'];
        $pass  = $form['passvalidacion'];
        $flag = 1;
    }else{
        $email = null;
        $pass = null;
        $flag = 0;
    }

    if($flag === 1){
        $hora_actual = horaActual();

        $conexion = conectarDB();
```

```
        $sentencia = "SELECT hash FROM usuarios WHERE `email`=?";

        if($stmt = mysqli_prepare($conexion, $sentencia)){
            // Agregamos la variable $email como parámetro a la sentencia
preparada
            mysqli_stmt_bind_param($stmt, 's', $email);
            if(!mysqli_stmt_execute($stmt)){
                validacionErrorEmailPass($conexion, $pathInclude);
            }else{
                // Almacenamos el resultado
                mysqli_stmt_store_result($stmt);
                // Vinculamos el resultado a una variable
                mysqli_stmt_bind_result($stmt, $hashedPassBD);
                // Obtenemos el resultado (hash de la contraseña de la
Base de Datos
                mysqli_stmt_fetch($stmt);
                // Si se verifica la contraseña introducida por el usuario
con su hash, proseguimos
                if(password_verify($pass, $hashedPassBD)){
                    $num_rows = mysqli_stmt_num_rows($stmt);
                    if($num_rows != 0){
                        // Obtenemos las horas y fechas de reserva del
usuario
                        $sentencia = "SELECT
time_to_sec(`hora_reserva`),fecha_reserva FROM `reserva` WHERE "
                                    . "`email` = ? AND `fecha_reserva` =
CURDATE() OR"
                                . "`fecha_reserva` = DATE_SUB(CURDATE(),
INTERVAL 1 DAY)";

                        if($stmt = mysqli_prepare($conexion, $sentencia)){
                            mysqli_stmt_bind_param($stmt, 's', $email);
                            if(!mysqli_stmt_execute($stmt)){
                                validacionErrorEmailPass($conexion,
$pathInclude);
                            }else{
                                // Almacenamos el resultado
                                mysqli_stmt_store_result($stmt);
                                // Vinculamos el resultado a una variable
                                mysqli_stmt_bind_result($stmt, $hora_lab,
$fecha_reserva_lab);
                                // Obtenemos las horas de reserva y
vinculamos con la que se ha accedido
                                while(mysqli_stmt_fetch($stmt)){
                                        $horas_reserva[] = $hora_lab;
                                        $fecha_reserva[] =
$fecha_reserva_lab;
                                }
                                // Comprobaciones para saber si la hora de
acceso es a las 00:00:00
                                // o pertenece al día actual
                                for($i = 0; $i < count($horas_reserva);
$i++){
                                    if((($horas_reserva[$i]+ $t_interfaz)
> $hora_actual) &&
                                        ($hora_actual >
$horas_reserva[$i]) && $fecha_reserva[$i] == date('Y-m-d'))){
                                        $hora_reserva_lab =
$horas_reserva[$i];
                                        break;
```

```
                                    }elseif(((($horas_reserva[$i]+
$t_interfaz) > $hora_actual) &&
                                            ($hora_actual >
$horas_reserva[$i]) &&
                                            ($fecha_reserva[$i] ==
date('Y-m-d', strtotime('-1 day'))) &&
                                            ($horas_reserva[$i] == 0)){
                                        $hora_reserva_lab = 0;
                                        break;
                                    }else{
                                        $hora_reserva_lab = $t_dia;
                                    }
                                }

                                // Comprobamos si la hora de reserva está
en el
                                // rango permitido para el acceso a la
interfaz.
                                // Sino, denegamos el acceso
                                if((($hora_reserva_lab + $t_interfaz) >
$hora_actual) &&
                                        ($hora_actual >
$hora_reserva_lab)){
                                    // Asignaciones para la variable de
sesion

                                    $_SESSION['usuario'] = $email;
                                    $_SESSION['pass'] = $pass;
                                    $_SESSION['hora_reserva'] =
$hora_reserva_lab;
                                    $_SESSION['timeout'] =
$hora_reserva_lab + $t_interfaz;
                                    $_SESSION['estado'] = 1;
                                    $_SESSION['id'] = session_id();
                                    $limite_sesion = $_SESSION['timeout']
- horaActual();
                                    $limite_sesion_ms = $limite_sesion *
1000;

                                    // Marcamos como accedido
                                    $accedido = 1;
                                    $sentencia_2 = "UPDATE hctlab.reserva
SET `acceso`=? WHERE `email` =? "
                                            . "AND
`hora_reserva`=sec_to_time(?) AND `fecha_reserva` = CURDATE()";
                                    if($stmt = mysqli_prepare($conexion,
$sentencia_2)){
                                        mysqli_stmt_bind_param($stmt,
'iss', $accedido, $email, $hora_reserva_lab);

                                        if(!mysqli_stmt_execute($stmt)){
                                            echo "Error en la Base de
Datos <br />";
                                            echo "Inténtelo de nuevo <br
/><br />";
                                            echo '<a class="button-fail
button-fail-orange pure-button"'
                                                    . ' href = "' .
$pathInclude . 'inicio.php">Regresar<br /></a>';
                                            mysqli_close($conexion);
                                        }else{ ?>
                            <!--Subimos los ficheros -->
```

```
                            <div class="pure-form ">
                                <form action = "upload_1.php" method =
"post" enctype = "multipart/form-data">
                                                    ¿Necesita la librería
ieee_proposed (fixed)?
                                    <input id= "radio-1" class="radio-lab"
name="libreria" type="radio" value = "1">
                                    <label for= "radio-1" class="radio-
lab-label">No</label>
                                    <input id= "radio-2" class="radio-lab"
name="libreria" type="radio" value = "2" checked>
                                    <label for= "radio-2" class="radio-
lab-label">Sí</label><br /> <br />
                                    <label for = "archivo"> Archivos .vhd
a cargar </label>
                                    <i class="fa fa-upload fa-2x"></i>
                                    <input type="file" multiple="multiple"
name="archivo[]" id="archivo"/><br /><br />
                                    <button type ="submit" class="button-
success button-success-green pure-button" ><b>Enviar</b></button><br /><br
/>
                                        La compilación de los ficheros y
programación de la FPGA puede durar 2/3 minutos.<br />
                                                        Por favor,
manténgase a la espera.<br />
                                </form>
                            `</div> <?php
                                        }
                                    }else{
                                        echo "Error en la Base de Datos
<br />";
                                        echo "Inténtelo de nuevo <br /><br
/>";
                                        echo '<a class="button-fail
button-fail-orange pure-button"'
                                                . ' href = "' .
$pathInclude . 'inicio.php">Regresar<br /></a>';
                                        mysqli_close($conexion);
                                    }
                                }else{
                                    validacionErrorHoraReserva($conexion,
$pathInclude);
                                }
                            }
                        }else{
                             validacionErrorEmailPass($conexion,
$pathInclude);
                        }
                    }else{
                        validacionErrorEmailPass($conexion, $pathInclude);
                    }
                }else{
                    validacionErrorEmailPass($conexion, $pathInclude);
                }
            }
        }else{
            echo "Error en la Base de Datos <br />";
            echo "Inténtelo de nuevo <br />";
            echo '<a class="button-fail button-fail-orange pure-button"'
                    . ' href = "' . $pathInclude .
'inicio.php">Regresar<br /></a>';
```

```
            mysqli_close($conexion);
        }
    }else{
        validacionErrorDatosForm($pathInclude);
    } ?>
        </div>
    </div>
</div>
 <div class="content-wrapper">
    <div class="content">
        <!--Uso de Javascript para cerrar sesión en caso de que se alcance
                el límite de tiempo en uso de la sesión iniciada-->
        <script type= "text/javascript">
            // Captura de la variable que en el código PHP calculaba los
ms restantes
            var milisegundos = <?php echo $limite_sesion_ms;?>;
            // Si se ha alcanzado el límite, cerramos sesión
automáticamente
            // invocando la página de cerrarsesion.php
            function killerSession(){
                setTimeout("window.open('cerrarsesion.php','_top');",
milisegundos);
            }
            // Llamada a la función para cerrar la sesión después de que
toda la página se ha cargado
            window.onload = function(){
                killerSession();
            };
        </script>
<?php include("/include/pie.inc"); ?>
```

II.2.12) upload_1.php

```
<!-- upload_1.php -->
<?php session_start();
      include("/include/cabeceraLab.inc");
      include("/include/funciones.php");?>

<div class="cont-ppal-upload_1">
    <div class="cont-upload_1">
        <div class="cont-cabecera-upload_1">
<?php
        if($_POST['libreria'] == 1){
            $libreria = 0; // El alumno ha seleccionado que no necesita la
librería ieee_proposed
        }elseif($_POST['libreria'] == 2){
            $libreria = 1; // Sí necesita la librería ieee_proposed
        }
        // Borramos posibles archivos o directorios creados anteriormente
        borrarDirAlumno();
        // Comprobamos si el fichero que ha subido es válido para nuestra
interfaz
        $val_manejo_archivo = manejoArchivo($libreria);
        $hora_actual = horaActual();

        $limite_sesion = $_SESSION['timeout'] - horaActual();
        $limite_sesion_ms = $limite_sesion * 1000;

        if(isset($_SESSION['estado']) && $_SESSION['estado'] == 1
```

```
                && (($_SESSION['hora_reserva'] + $t_interfaz) >
$hora_actual)){
            //Si ha habido algún error con el fichero, regreso a cargar
uno nuevo
            if($val_manejo_archivo === 0){
                $_SESSION['estado'] = 2;
                ?><form action ="upload.php" method = "post" enctype =
"multipart/form-data"><br /><br />
                    <button type ="submit" class="button-fail button-fail-
orange pure-button"><b>Regresar</b></button>
                  </form><?php
            }else{
            //Si no hay errores con el fichero subido, prosigo a mostrar
la interfaz
            // Configuramos los parámetros para la comunicación del puerto
            $execPuerto = 'mode ' . $puertoCOM .' baud=19200 data=8 stop=1
parity=n xon=on';
            exec($execPuerto); ?>
            <div class= "nueva">
            <form action ="interfazLab.php" method = "post" enctype =
"multipart/form-data"><br /><br /><br />
                        Seleccione los valores con los que inicialmente
quiere programar la FPGA:<br />
                    <input id= "radio-1" class="radio-lab" name="swap"
type="radio" value = "1">
                    <label for= "radio-1" class="radio-lab-label">Swap =
0</label>
                    <input id= "radio-2" class="radio-lab" name="swap"
type="radio" value = "2" checked>
                    <label for= "radio-2" class="radio-lab-label">Swap =
1</label>
                    <input id= "radio-3" class="radio-lab"
name="reset_total" type="radio" value = "1" checked>
                    <label for= "radio-3" class="radio-lab-label">Reset =
0</label>
                    <input id= "radio-4" class="radio-lab"
name="reset_total" type="radio" value = "2">
                    <label for= "radio-4" class="radio-lab-label">Reset =
1</label><br /><br />
                    <label for= "slider">Consigna</label>
                    <input type= "range" id="slider" name="consigna"
min="0" max="100" step="10" value="50">
                    <br /><br /><div id="mostrar_consigna"></div>
                    <input type= "hidden" name="val_consigna" id=
"val_consigna" value=""><br />
                    <button type ="submit" class="button-success button-
success-green pure-button"><b>Continuar</b></button>
                </form><?php
            }
        }else{
            validacionErrorHoraReserva($pathInclude);
        } ?>
            </div>
        </div>
    </div>
</div>
 <div class="content-wrapper">
    <div class="content">
        <!-- Script para capturar el valor de la consigna y
             mostrarlo según se cambie de valor-->
        <script type="text/javascript">
```

```
    $(document).ready(function(){
            $(function() {
                $("#slider").slider();
            });
            $('#slider').change(function(){
                $('#val_consigna').val($(this).val());
                $('#mostrar_consigna').text($(this).val());

            });
            $('#val_consigna').val($('#slider').val());
            $('#mostrar_consigna').text($('#slider').val());
    });
        </script>

        <!--Uso de Javascript para cerrar sesión en caso de que se alcance
                el límite de tiempo en uso de la sesión iniciada-->
        <script type= "text/javascript">
            // Captura de la variable que en el código PHP calculaba los
ms restantes
            var milisegundos = <?php echo $limite_sesion_ms;?>;
            // Si se ha alcanzado el límite, cerramos sesión
automáticamente
            // invocando la página de cerrarsesion.php
            function killerSession(){
                setTimeout("window.open('cerrarsesion.php','_top');",
milisegundos);
            }
            // Llamada a la función para cerrar la sesión después de que
toda la página se ha cargado
            window.onload = function(){
                killerSession();
            };
        </script>
<?php include("/include/pie.inc"); ?>
```

II.2.13) cabeceraLab.inc

```
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8"
/>
        <meta name="viewport" content="width=device-width, initial-
scale=1">

        <link rel="stylesheet" href="http://netdna.bootstrapcdn.com/font-
awesome/4.0.3/css/font-awesome.css">
        <link rel="stylesheet"
href="http://yui.yahooapis.com/pure/0.6.0/grids-responsive-min.css">
        <link rel="stylesheet"
href="http://yui.yahooapis.com/pure/0.6.0/pure-min.css">
        <script src="https://code.jquery.com/jquery-
1.11.3.min.js"></script>
        <script src="//code.jquery.com/ui/1.11.4/jquery-ui.js"></script>
        <link rel="stylesheet"
href="//code.jquery.com/ui/1.11.4/themes/smoothness/jquery-ui.css">
        <link rel="stylesheet" href="/labRemoto/include/style.css">
        <title>Laboratorio Sistemas de Control</title>
```

```
    </head>

    <body>
    <div class="header">
        <div class="home-menu pure-menu pure-menu-horizontal pure-menu-
fixed">
            <a class="pure-menu-heading" href="inicio.php">Inicio</a>
            <a class="pure-menu-heading" href="calendario.php">Acceso a
calendario</a>
            <a class="pure-menu-heading" href="borrado.php">Borrar
reserva</a>
            <a class="pure-menu-heading" href="validacion.php">Acceso a
interfaz</a>
            <a class="pure-menu-heading button-fail button-fail-red pure-
button" href="cerrarsesion.php">Cerrar Sesión</a>
        </div>
    </div>
```

II.2.14) cabeceraLab_1.inc

```
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8"
/>
        <meta name="viewport" content="width=device-width, initial-
scale=1">

        <link rel="stylesheet" href="http://netdna.bootstrapcdn.com/font-
awesome/4.0.3/css/font-awesome.css">
        <link rel="stylesheet"
href="http://yui.yahooapis.com/pure/0.6.0/grids-responsive-min.css">
        <link rel="stylesheet"
href="http://yui.yahooapis.com/pure/0.6.0/pure-min.css">
        <script src="https://code.jquery.com/jquery-
1.11.3.min.js"></script>
        <link rel="stylesheet" href="/labRemoto/include/style_2.css">

        <title>Laboratorio Sistemas de Control</title>
        <script>
        function rellenaInfo() {
            $.ajax({
                    url: "/labRemoto/lecturaPuerto.php",
                    dataType: "html"
                }).done(function(msg) {
                    // Método que nos permite acceder a la referencia del
id mensaje,
                    // capturando los datos de los LEDs
                    document.getElementById("mensaje").innerHTML=msg;
                });
        }
            loop = setInterval(rellenaInfo,5000);
        </script>
    </head>

<body>
<div class="header">
    <div class="home-menu pure-menu pure-menu-horizontal pure-menu-fixed">
```

```
        <a class="pure-menu-heading" href="inicio.php">Inicio</a>
        <a class="pure-menu-heading" href="calendario.php">Acceso a
calendario</a>
        <a class="pure-menu-heading" href="borrado.php">Borrar reserva</a>
        <a class="pure-menu-heading" href="validacion.php">Acceso a
interfaz</a>
        <a class="pure-menu-heading button-success button-success-green
pure-button" href="upload.php">Subir nuevos ficheros</a>
        <a class="pure-menu-heading button-fail button-fail-red pure-
button" href="cerrarsesion.php">Cerrar Sesión</a>
    </div>
</div>
```

II.2.15) interfazLab.php

```
<!-- interfazLab.php -->
<?php session_start();
      include("/include/cabeceraLab_1.inc");
      include("/include/funciones.php");?>

<div class="cont-container-lab-prueba-29">
    <div class="cont-lab-prueba-29">
<?php
    $hora_actual = horaActual();
    // Si todavía no se ha superado la hora de uso de la interfaz,
mostramos la página
    if(isset($_SESSION['estado']) && (($_SESSION['hora_reserva'] +
$t_interfaz) > $hora_actual)){

        $_SESSION['estado'] = 2;
        $limite_sesion = $_SESSION['timeout'] - horaActual();
        $limite_sesion_ms = $limite_sesion * 1000;

        // Asignaciones para enviar los datos correctamente a la UART
        // Nos aseguramos de poner un int
        settype($_POST['consigna'], "integer");
        $int_consigna = $_POST['consigna']; ?>

        <iframe src="http://169.254.170.233" id ="anuncio"
style="width:70%; height:98%;"></iframe>
        <div class="pure-g">
            <div class="pure-u-1-24"></div>
                <div class="pure-u-11-24">
                    <?php echo "Valor actual de Consigna: " .
$int_consigna . "%";?>
                    <form action ="interfazLab.php" method = "post"
enctype = "multipart/form-data">
                        <!-- El valor 1 se corresponde con un 0 y el valor
2 con un 1 en cada caso-->
                        <input id= "radio-1" class="radio-lab" name="swap"
type="radio" value = "1">
                        <label for= "radio-1" class="radio-lab-label">Swap
= 0</label>
                        <input id= "radio-2" class="radio-lab" name="swap"
type="radio" value = "2" checked>
                        <label for= "radio-2" class="radio-lab-label">Swap
= 1</label>
                        <input id= "radio-3" class="radio-lab"
name="reset_total" type="radio" value = "1" checked>
```

```
                        <label for= "radio-3" class="radio-lab-
label">Reset = 0</label>
                        <input id= "radio-4" class="radio-lab"
name="reset_total" type="radio" value = "2">
                        <label for= "radio-4" class="radio-lab-
label">Reset = 1   </label>
                        <input type= "hidden" name="val_consigna" id=
"val_consigna" value="">
                        <input type="submit" value="Enviar"></div>
                    <div class="pure-u-11-24">
                        <label for=
"slider">Consigna   </label>
                        <input type= "range" id="slider" name="consigna"
min="0" max="100" step="10" value="50">
                    </form>

          

          
                    <!--Mostramos el valor capturado del slide-->
                    <div id ="mostrar_consigna"></div>
                    <!--Mostramos el valor del puerto leído-->
                    <div id="mensaje"></div>
                </div>
            <div class="pure-u-1-24"></div> <?php
        // Para que no supere el máximo valor ascii (255) ni en hex (ff)
        if($int_consigna === 100){
            $int_consigna = 99.9;
        }
        // Multiplico para poder obtener los números enteros con 8 bits
        $valor_consigna = 2.56256;
        $valor_int_consigna = $int_consigna * $valor_consigna;
        $consigna_ascii = chr($valor_int_consigna);

        if($_POST['swap'] == 1){
            $swap = 0;
        }elseif($_POST['swap'] == 2){
            $swap = 1;
        }
        $swap_ascii = chr($swap);
        if($_POST['reset_total'] == 1){
            $reset = 0;
        }elseif($_POST['reset_total'] == 2){
            $reset = 1;
        }
        $reset_ascii = chr($reset);

        // Contador para intentar abrir el puerto varias veces antes de
que de error
        $i = 0;
        do{
            $fp = fopen($puertoCOM, "w+");
            $i++;
        }while(!$fp && $i < 5);

        if(!$fp){
            echo "Imposible abrir puerto <br />";
            borrarDirAlumno();
            echo '<a class="button-fail button-fail-orange pure-button"
href = "'
            .$pathInclude . 'inicio.php">Regresar<br /></a>';
```

```
        }else{
            // Enviamos por el puerto hacia la FPGA para programarla con
los
            // valores que ha escogido el alumno
            $res = fputs($fp, $swap_ascii);
            $res_1 = fputs($fp, $consigna_ascii);
            $res_2 = fputs($fp, $reset_ascii);
            fclose($fp); // Antes estaba abajo
        }
    }else{
        borrarDirAlumno();
        header('location: '. $pathInclude. 'cerrarSesion.php');
    } ?>
        </div>
        <!-- Script para capturar el valor de la consigna y
             mostrarlo según se cambie de valor-->
        <script type="text/javascript">
    $(document).ready(function(){
            $(function() {
                $("#slider").slider();
            });
            // Captura el valor al que se cambia
            $('#slider').change(function(){
                $('#val_consigna').val($(this).val());
                $('#mostrar_consigna').text($(this).val());

            });
            // Mostramos el valor actualizado
            $('#val_consigna').val($('#slider').val());
            $('#mostrar_consigna').text($('#slider').val());
    });
        </script>
        <!--Uso de Javascript para cerrar la ventana en caso de que se
alcance
                el límite de tiempo en uso de la sesión iniciada-->
        <script type= "text/javascript">
            // Captura de la variable que en el código PHP calculaba los
ms restantes
            var milisegundos = <?php echo $limite_sesion_ms;?>;
            // Si se ha alcanzado el límite, cerramos sesión
automáticamente
            // invocando la página de cerrarsesion.php
            function killerSession(){
                setTimeout("window.open('cerrarsesion.php','_top');",
milisegundos);
            }
            // Llamada a la función para cerrar la sesión después de que
toda la página se ha cargado
            window.onload = function(){
                killerSession();
            };
        </script>
    </div>
</div>
</body>
</html>
```

II.2.15) lecturaPuerto.php

```
<!-- lecturaPuerto.php -->
<?php
    session_start();
    include("/include/funciones.php");

    // Comprobamos si se ha acabado el tiempo máximo de la sesión
    $limite_sesion = $_SESSION['timeout'] - horaActual();

    if($limite_sesion == 0){
       cerrarSesionAlumno();
       header('location: '. $pathInclude. 'cerrarSesion.php');
    }else{
        // Obtenemos la ruta del ejecutable para leer el puerto
        list($pathname, $pathnameVHD, $pathLib_remota, $myDir,
$pathXilinx, $pathReadUART) = asignacionRutas();
        // Ejecutamos en cmd y salvamos el resultado en un array
        exec($pathReadUART, $readUART);
        // Convertimos a binario
        $readUARTLEDs = decbin($readUART[0]);
        // Si no obtenemos nada, mostramos error, en caso contrario,
        //devolvemos el valor obtenido del array
        if($readUART == null){
            echo "Error en la lectura de los LEDs <br />";
        }else{
            echo "El valor de los LEDs es: ";
            echo $readUART[0]; ?><sub>(10)</sub><?php
            echo $readUARTLEDs;?><sub>(2)</sub><?php
        }
    }
?>
```

II.2.15) funciones.php

```
<!-- funciones.php -->
<?php
// Definicion de variables globales

// Tiempo máximo de uso de la interfaz para el alumno (en segundos)
global $t_interfaz;
       $t_interfaz = 1200; // 20 min
// Tiempo máximo de permanencia de espera para nueva reserva (en segundos)
global $t_reserva;
       $t_reserva = 7200;  // 2 horas
// Número de segundos en un día
global $t_dia;
       $t_dia = 86400;
// Asignación para la comprobación de permanencia en la reserva teniendo
en cuenta
// el cambio de día
global $t_reserva_fin_dia;
       $t_reserva_fin_dia = $t_dia - $t_reserva;

// Directorio donde se encuentran alojadas las páginas php
global $pathInclude;
       $pathInclude =(string)"/labRemoto/";
// Puerto COM donde se instala el FTDI para conversión USB-UART
global $puertoCOM;
```

```
        $puertoCOM = 'COM8:';

// Definición de constantes para la BD
define("MAX_NUM_PASS", 99999999);
// Alojamiento al que nos conectamos
define("HOST", "localhost");
// Nombre de usuario de la Base de Datos
define("USER", "epsuam");
// Contraseña de la Base de Datos
define("PASSWORD", "xxxxxxxx");
// Nombre de la Base de Datos utilizada
define("DATABASE", "hctlab");

function asignacionRutas(){

    // Creamos un array para listar todas las rutas predefinidas
    $path = [];

    if(isset($_SESSION['usuario'])){
        // 1) Ruta para la creación del proyecto de cada alumno
        $path[] = "D:/XAMPP/htdocs/labRemoto//". $_SESSION['usuario'];
        // 2) Ruta donde se guardarán los ficheros que nos envía el alumno
        $path[] = "D:/XAMPP/htdocs/labRemoto/". $_SESSION['usuario'] .
"/files/";
        // 3) Ruta de la librería ieee_proposed remota para compilar
        $path[] = 'D:\Prueba_TFG\ieee_proposed\\';
        // 4) Ruta donde almacenamos los ficheros de nuestro servidor
necesarios
        // para la compilación y programación de la FPGA
        $path[] = "D:/Prueba_TFG/";
        // 5) Ruta donde está instalado Xilinx
        $path[] = 'D:\Xilinx_2014\14.4\ISE_DS\ISE\bin\nt64';
        // 6) Ruta donde está el ejecutable para leer el puerto
        $path[] = 'D:\Prueba_TFG\UART_2.exe';

        return $path;
    }else{
        return 0;
    }
}

// Convertimos los carácteres ASCII a decimal para poder mostrarlos
correctamente al alumno
function ascii_to_dec($str){
  $dec_array = [];
  echo $str."<br />";
  for ($i = 0, $j = strlen($str); $i < $j; $i++) {
    $dec_array[$i] = ord($str[$i]);
  }
  return $dec_array;
}

function manejoArchivo($libreria){

    list($pathname, $pathnameVHD, $pathLib_remota, $myDir, $pathXilinx,
$pathReadUART) = asignacionRutas();

    // Variable para comprobación de errores devueltos por la compilación
    $flag = [];
    // Variables para tener listas de los ficheros que se han subido
```

```
    $pathFileArray = []; // Directorio del fichero completo
    $nombreArray = [];   // Nombre de los ficheros
    $crearDir = 1; // Variable para el control de creacción del directorio
del alumno

    for($i = 0; $i < count($_FILES['archivo']['name']); $i++) {

        // Obtenemos únicamente el nombre del fichero, sin la ruta
        $nombre = basename($_FILES['archivo']['name'][$i]);
        $nombre_tmp = $_FILES['archivo']['tmp_name'][$i];

        // Nombre del archivo subido que se copiará
        $nombre_unico = basename($_FILES['archivo']['name'][$i], ".vhd");

        // Solo permitimos subir archivos .vhd
        $extension_permitida = array('vhd');
        // Extensiones extra no permitidas
        $extensiones_invalidas = array('php','php3','php4',
'php5','phtml','exe');
        // Caracteres permitidos en el nombre del fichero (incluyendo _ y
-)
        $caracteres_validos = '/[A-Za-z0-9_\-]/';
        // Nos quedamos con la extensión del archivo
        $extension = end(explode('.',
basename($_FILES['archivo']['name'][$i])));
        // Comprobamos si se encuentra la extensión permitida
        $extension_ok = in_array($extension, $extension_permitida);

        // Comprobamos que se ha enviado un archivo
        if($_FILES['archivo']['size'][$i] == 0){
            echo 'No se han seleccionado archivos<br />';
            return 0;
        // Comprobamos que no contiene extensiones
        }elseif(in_array(end(explode('.',
basename($_FILES['archivo']['name'][$i]))), $extensiones_invalidas)){
            echo "Extensión no permitida.<br />";
            return 0;
        }elseif(preg_match($caracteres_validos,
basename($_FILES['archivo']['name'][$i]))!= 1){
                echo "Nombre de archivo inválido. Renómbrelo de otra
manera.<br />";
                return 0;
            if(preg_match("/exe/",
basename($_FILES['archivo']['name'][$i]) == 1)){
                echo "Nombre de archivo inválido. Renómbrelo de otra
manera.<br />";
                return 0;
            }

        }elseif( $_FILES['archivo']['error'][$i] > 0){

             switch($_FILES['archivo']['error'][$i]){
                case 1:     // UPLOAD_ERR_INI_SIZE
                case 2:     // UPLOAD_ERR_FORM_SIZE
                    echo 'Tamaño máximo permitido. Suba de nuevo un
archivo válido';
                    break;
                case 3:     // UPLOAD_ERR_PARTIAL
                    echo 'Archivo subido parcialmente. Suba de nuevo un
archivo válido';
                    break;
```

```
                case 4:     // UPLOAD_ERR_NO_FILE
                    echo 'No se ha subido archivo. Suba uno';
                    break;
                case 6:     // UPLOAD_ERR_NO_TMP_DIR
                    echo 'Directorio temporal no encontrado Suba de nuevo
un archivo';
                    break;
                case 7:     // UPLOAD_ERR_CANT_WRITE
                    echo 'Error al escribir en disco. Suba de nuevo un
archivo';
                    break;
                case 8:     // UPLOAD_ERR_EXTENSION
                    echo 'Una extensión de PHP paró la subida. Suba de
nuevo un archivo';
                    break;
                default :
                    echo 'Error en archivo. Suba uno de nuevo';
                    break;
            }
            return 0;
        }elseif($extension_ok){

            if($crearDir == 1){
                if(!mkdir($pathnameVHD, 0777, true)){
                    echo "Error al crear la carpeta del alumno <br />";
                    return 0;
                }else{
                    // Evitamos que se vuelva a crear el directorio
                    // tras la comprobación de los siguientes archivos
                    $crearDir = 0;
                }
            }
            // Comprobamos que se ha subido el archivo y lo copiamos al
directorio creado
            if(is_uploaded_file($nombre_tmp)){
                if(!move_uploaded_file($nombre_tmp, $pathnameVHD .
$nombre_unico )){
                    echo "No ha sido posible copiar los archivos.<br />
Inténtelo de nuevo";
                    return 0;
                }
            }else{
                echo "Error en el fichero subido. Inténtelo de nuevo<br
/>";
                return 0;
            }
             //Ruta y nombre de cada archivo almacenado
             $pathFile = $pathnameVHD . $nombre_unico;
             $pathFileArray[] = $pathFile;
             $nombreArray[] = $nombre_unico;
             // Compilamos cada uno de los archivos subidos
             $res_compilacion = compilacionVHD($pathname, $pathnameVHD,
$pathFile, $pathLib_remota, $nombre_unico, $libreria);

        }else{
             echo "Archivo inválido. Seleccione archivo <b>.vhd</b>
correcto";
             return 0;
        }
```

```
        if($res_compilacion === 0){
            $flag[] = 0;
        }else{
            $flag[] = 1;
        }
    }

    for($i = 0; $i < count($flag); $i++){
        if($flag[$i] === 0){
            echo "<br /> <br /> Ha habido errores de compilación en los
archivos subidos.<br / >";
            echo "Revíselos y súbalos de nuevo";
            errorGeneracionBit($pathname, $pathnameVHD);
            return 0;
        }else{
            continue;
        }
    }
    $resGeneracionBit = generacionBit($pathname, $pathnameVHD,
$pathFileArray, $nombreArray, $myDir, $pathLib_remota, $libreria);
    if($resGeneracionBit === 0){
        errorGeneracionBit($pathname, $pathnameVHD);
        return 0;
    }else{

        $resProgramarFPGA = programarFPGA($pathname, $pathnameVHD,
$pathFileArray, $pathXilinx);
        if($resProgramarFPGA === 0){
            errorGeneracionBit($pathname, $pathnameVHD);
            return 0;
        }else{
            return 1;
        }
    }
    return 1;
}

function compilacionVHD($pathname, $pathnameVHD, $pathFile,
$pathLib_remota, $nombre_unico, $libreria){


        // Comandos a ejecutar con compilación de la librería necesaria
        $path = [];
        //$path[] = 'run modelsim'; MODELSIM debe estar iniciado
        if($libreria == 0){
            $path[] = 'vlib work';
            $path[] = 'vcom -reportprogress 300 -work work ' . $pathFile;
        }elseif($libreria == 1){
            $path[] = 'vlib work';
            $path[] = 'vlib ieee_proposed';
            $path[] = 'vcom -reportprogress 300 -work ieee_proposed ' .
                    $pathLib_remota . 'fixed_float_types_c.vhdl';
            $path[] = 'vcom -reportprogress 300 -work ieee_proposed ' .
                    $pathLib remota . 'fixed pkg c.vhdl';
            $path[] = 'vcom -reportprogress 300 -work ieee_proposed ' .
                    $pathLib_remota . 'fixed_synth_c.vhdl';
            $path[] = 'vcom -reportprogress 300 -work work ' . $pathFile;
        }else{
            echo "Ha habido un error en la compilación y las librerías.";
            $i = 0;
            return $i;
```

```
        }


        for($i = 0; $i < count($path); $i++){
            $pathflow = $path[$i] . ' >>' . $pathnameVHD
.'compilacion.txt';
            $pathflow = addslashes($pathflow);
            system($pathflow);
        }
        // Ruta del fichero con los resultados de log de la compilación
        $pathCompilacion = $pathnameVHD ."compilacion.txt";
        $i = 0;
        $file = fopen($pathCompilacion, "r");
        if(!$file){
            echo "Ha habido un error en la compilación.<br />Suba los
ficheros de nuevo.";
            return $i;
        }else{
            $i = 1;

            while(!feof($file)){
                while($line = fgets($file)){

                    // Buscamos posibles comandos de error en la salida de
compilación.
                    // Si hay coincidencia, informamos al usuario de dónde
está exactamente el error.
                    $error_compilacion = preg_match('/Error:/', $line);
                    if($error_compilacion){
                        $line_2 = preg_split('/\(/', $line);
                        $line_3 = preg_split('/\) /', $line);
                        //
                        if(isset($line_3[1])){
                            echo "Error en: " . $nombre_unico . "(".
$line_2[1] . $line_3[1] . "<br />";
                        }
                        echo "Error en: " . $nombre_unico . "(".
$line_2[1] ."<br />";
                        $i = 0;
                    }
                }
            }
        }
    fclose($file);
    // Una vez impresos los errores por pantalla borramos el fichero para
    // evitar sobreescribirlo en caso de subir más archivos .vhd
    unlink($pathCompilacion);
    // También borramos todos los ficheros subidos
    //system('quit -force');
    return $i;
}

function generacionBit($pathname, $pathnameVHD, $pathFileArray,
$nombreArray, $myDir, $pathLib remota, $libreria){

    $pathTopPrj = $pathnameVHD . 'top.prj';
    $file = fopen($pathTopPrj, "w+");
    if($file === false){
        echo "Error al crear Top.prj. Regrese y suba sus ficheros de
nuevo";
```

```
            return 0;
        }else{

            $pathFileArray[] = $pathTopPrj;

            // Método para obtener el nombre de los ficheros que entrego al
alumno
            // sin el directorio donde estén ubicados
            $myDirFilesArray = array_diff(scandir($myDir), array(".", "..") );
            if(!$myDirFilesArray){
                echo "No se han podido copiar los ficheros para programar la
FPGA. <br />";
                echo "Regrese a subir ficheros. <br />";
                unlink($pathTopPrj);
                return 0;
            }
            // Copiamos los ficheros propios necesarios para la creación del
.bit
            foreach($myDirFilesArray as $value){
                // Comprobamos que los ficheros que copiemos no tengan el
mismo nombre que alguno
                // que haya subido el alumno. Si coinciden, renombramos
nuestro fichero
                // En caso de que no se pueda, pedimos renombrar al usuario el
suyo
                foreach($nombreArray as $value_2){
                    if(strcmp($value, $value_2) == 0){

                        $extension = explode('.', $value);
                        $value = $extension[0] . '_new.vhd';
                        if(!($renombre = rename($myDir . $value_2, $myDir .
$value))){
                            echo "No se ha podido renombrar un fichero que
sobreescribía a uno suyo.<br />";
                            echo "Asigne otro nombre a ". $value_2;
                            return 0;
                        }else{
                         continue;
                        }
                    }
                }

                if(!copy($myDir . $value, $pathnameVHD . $value)){
                    echo "Error al copiar los ficheros necesarios para generar
.bit . <br />";
                    echo "Regrese y súbalos de nuevo. <br /> ";
                    return 0;
                }else{
                    $pathFileArray[] = $pathnameVHD . $value;
                    $nombreArray[] = $value;
                }
            }

            // Escribimos en el top.prj la librería y todos los ficheros
presentes en el
            // directorio necesarios para la generación del .bit
            if($libreria == 1){
                fwrite($file, "vhdl ieee_proposed " . "\"" . $pathLib_remota .
"fixed_float_types_c.vhdl" . "\"" ."\r\n");
                fwrite($file, "vhdl ieee_proposed " . "\"" . $pathLib_remota .
"fixed_pkg_c.vhdl" . "\"" ."\r\n");
```

```
            fwrite($file, "vhdl ieee_proposed " . "\"" . $pathLib_remota .
"fixed_synth_c.vhdl" . "\"" ."\r\n");
        }
        // En el caso del fichero .ucf lo omitimos en el Top.prj, pues
dará problemas de sintetización
        $extension_no_permitida = array('ucf');

        foreach($nombreArray as $value){
            // Nos quedamos con la extensión del fichero
            $extension = end(explode('.', $value));
            // Comprobamos si se encuentra la extensión no permitida
            $extension_ok = in_array($extension, $extension_no_permitida);
            // Si no ha devuelto true, copiamos el nombre del fichero en
Top.prj
            if(!$extension_ok){
                $line = addslashes("$value");
                fwrite($file, "vhdl work " . "\"" . $line . "\"" ."\r\n");
            }
        }
        fclose($file);
        return 1;
    }
    return 0;
}

function errorFiles($pathFileArray){

    foreach($pathFileArray as $value){
        if(file_exists($value)){
            unlink($value);
        }else{
            echo "Error al borrar los ficheros que ha subido. Regrese al
inicio"; //MIRAR QUÉ HAGO EN ESTE CASO!
        }
    }
}

function programarFPGA($pathname, $pathnameVHD, $pathFileArray,
$pathXilinx){

    chdir($pathXilinx);
    // Llamo a xflow para poder ejecutar desde línea de comandos y generar
el .bit
    // Los posibles errores y ejecución en consola se extraen a ficheros
.txt
    $pathXflow = "xflow -p  xc3s1000ft256-4 -implement balanced.opt -
config bitgen.opt -wd " .
            "\"" . $pathnameVHD . "\"" . " -synth xst_vhdl.opt top.prj";

    $pathXflow_volcado = $pathXflow . ' >' . $pathname .'\\out.txt 2>' .
$pathname .'\\errores.txt &';

    $pathXflow_esc = addslashes($pathXflow_volcado);
    system($pathXflow esc);

    $file = fopen($pathname . '\\out.txt', "r");
    if($file === false){
        echo "Error al sintetizar el diseño y generar .bit";
        return 0;
    }else{
```

```
        while(!feof($file)){
            while($line = fgets($file)){
                // Buscamos el comando de xflow done! que indica que se ha
                // finalizado con éxito la generación del .bit
                $xflowDone = preg_match('/xflow done\!/', $line);
                if($xflowDone){
                    echo $line . "<br />" . "Generación del fichero .bit
realizada con éxito.<br />";
                    echo "Procediendo a programación de la FPGA . . .";
                    // Función que genera fichero .cmd con los comandos
para programar la FPGA con el .bit
                    $resComandosImpact =
generacionComandosImpact($pathnameVHD);
                    if($resComandosImpact === 0){
                        return 0;
                    }else{
                        $pathCMD= $pathnameVHD . 'comandos.cmd';
                        $pathImpact = 'impact -batch ' . '"' . $pathCMD .
'"';
                        $pathImpact_volcado = $pathImpact . ' >' .
$pathname .'\\out_impact.txt 2>' . $pathname .'\\errores_impact.txt &';
                        $pathImpact_esc = addslashes($pathImpact_volcado);

                        system($pathImpact_esc);
                        $file_2 = fopen($pathname . '\\out_impact.txt',
"r");
                        if($file_2 === false){
                            echo "Error al ejecutar iMPACT <br />";
                            return 0;
                        }else{
                            while(!feof($file_2)){
                                while($line_2 = fgets($file_2)){

                                $impactError_1 = preg_match('/Cable
connection failed/', $line_2);
                                $impactError_2 = preg_match('/Cable
autodetection failed/', $line_2);
                                if($impactError_1 || $impactError_2){
                                    echo "Error en el cable de conexión de
la FPGA. <br />";
                                    echo "Informe al administrador para
seguir usando la interfaz correctamente <br />";
                                    fclose($file_2);
                                    return 0;
                                }
                                // Si se ha programado correctamente,
buscamos que así se ha escrito en el fichero
                                $impactDone = preg_match('/Programmed
successfully/', $line_2);
                                    if($impactDone){
                                        echo "La FPGA se ha programado
correctamente. <br />";
                                        echo "Proceda a poner los
parámetros.<br /><br />";
                                        fclose($file_2);
                                        return 1;
                                    }
                                }
                            }
                        }
                        fclose($file_2);
```

```
                    }
                    return 1;
                }
            }
        }
    }

    fclose($file);
    echo "No se ha podido generar el .bit. <br /> Revise sus ficheros e
inténtelo de nuevo.<br />";
    return 0;

}

function generacionComandosImpact($pathnameVHD){

    $pathCMD= $pathnameVHD . 'comandos.cmd';
    $file = fopen($pathCMD, "w+");
    if($file === false){
        echo "Error al crear fichero de comandos. Regrese al inicio";
        return 0;
    }else{
        $comandos = [];
        $comandos[] = 'setMode -bs';
        $comandos[] = 'setCable -p auto';
        $comandos[] = 'identify';
        $comandos[] = 'assignFile -p 1 -file ' . '"' . $pathnameVHD .
'top.bit' . '"';
        $comandos[] = 'program -p 1';
        // Escribimos los comandos
        for($i = 0; $i < count($comandos); $i++){
            if(!fwrite($file, $comandos[$i] ."\r\n")){
                echo "Error al generar comandos para programar la FPGA.";
                return 0;
            }
        }
    }
    fclose($file);
    return 1;
}

function errorGeneracionBit($pathname, $pathnameVHD){

    // Si existe el directorio, borramos su contenido hasta vaciar todo el
contenido de las carpetas
    if(file_exists($pathname)){
        foreach(glob($pathnameVHD . "/*") as $archivos_carpeta)
        {
            if (is_dir($archivos_carpeta)){
                errorGeneracionBit($pathname, $archivos_carpeta);
            }else{
                unlink($archivos_carpeta);
            }
        }
    }

    if(file_exists($pathnameVHD)){
        rmdir($pathnameVHD);
    }
}
```

```
//<!--validacion.php, upload.php y upload_1.php -->

function validacionErrorDatosForm($pathInclude){
    echo "Error. El usuario o contraseña no tienen el formato adecuado.<br
/>";
    echo "Inténtelo de nuevo o contacte con el administrador<br /><br />";
    echo '<a class="button-fail button-fail-orange pure-button" href = "'
. $pathInclude . 'inicio.php">Regresar<br /></a>';
}
function validacionErrorEmailPass($conexion, $pathInclude){
    echo "Error. El usuario o contraseña no se encuentran en la BD.<br />"
. mysqli_connect_error();
    echo "Inténtelo de nuevo o contacte con el administrador<br /><br />";
    echo '<a class="button-fail button-fail-orange pure-button" href = "'
. $pathInclude . 'inicio.php">Regresar<br /></a>';
    mysqli_close($conexion);
}

function validacionErrorHoraReserva($conexion, $pathInclude){
    echo "La hora de reserva no ha podido verificarse en un rango
válido.<br />" . mysqli_connect_error();
    echo "Vuelva a entrar a la hora que reservó o haga una nueva
reserva.<br /><br />";
    echo '<a class="button-fail button-fail-orange pure-button" href = "'
. $pathInclude . 'inicio.php">Regresar<br /></a>';
    mysqli_close($conexion);
}
//<!--calendario.php-->

function calendarioErrorSelReserva($stmt, $conexion, $pathInclude){
    echo "Error. No se ha podido comprobar la hora de reserva en la BD.<br
/>" .
    mysqli_connect_error();
    echo "Inténtelo de nuevo o contacte al administrador";
    echo '<a href = "' . $pathInclude . 'calendario.php"> <br /> Regresar
<a/>';
    mysqli_stmt_close($stmt);
    mysqli_close($conexion);
}

// <!--reserva.php-->

function reservaErrorUsuarioBD($conexion, $pathInclude){
    echo "Error. El usuario introducido no se encuentra en la BD " .
mysqli_connect_error();
    echo "Inténtelo de nuevo o contacte con el administrador de una vez<br
/><br />";
    echo '<a class="button-fail button-fail-orange pure-button" href = "'
. $pathInclude . 'calendario.php">Regresar<br /></a>';
    mysqli_close($conexion);
}

function reservaErrorUsuario($stmt, $conexion, $pathInclude){
    echo "Error. Los datos no son correctos.<br />";
    echo "Inténtelo de nuevo o contacte con el administrador<br /><br />";
    echo '<a class="button-fail button-fail-orange pure-button" href = "'
. $pathInclude . 'calendario.php">Regresar<br /></a>';
    mysqli_stmt_close($stmt);
    mysqli_close($conexion);
}
```

```
function reservaErrorUsuarioPass($pathInclude){
    echo "Error. El nombre de usuario o contraseña son incorrectos.<br
/>";
    echo "Inténtelo de nuevo o contacte con el administrador<br /><br />";
    echo '<a class="button-fail button-fail-orange pure-button" href = "'
. $pathInclude . 'calendario.php">Regresar<br /></a>';
}

// <!--reservaOk.php-->

function reservaOkErrorHoraActual($pathInclude){
    echo "Error. La hora introducida para reservar es menor que la hora
actual <br />"
    . "o pertenece al día siguiente.<br /><br /> ";
    echo "Elija una hora de reserva válida.<br /><br />";
    echo '<a class="button-fail button-fail-orange pure-button" href = "'
. $pathInclude . 'calendario.php">Regresar<br /></a>';
}

function reservaOkErrorAlmacenar($conexion, $pathInclude){
    echo "Error al almacenar la reserva";
    echo "Inténtelo de nuevo o contacte con el administrador.<br />";
    echo '<a class="button-fail button-fail-orange pure-button" href = "'
. $pathInclude . 'inicio.php">Regresar<br /></a>';
    mysqli_close($conexion);
}

function reservaOkErrorEstaReservado($stmt, $conexion, $pathInclude,
$t_reserva){
    echo "Error. No puede reservar a la hora que ha seleccionado."
    . "Tiene ya realizada una reserva en las " . $t_reserva/3600 . "
próximas horas.<br />";
    echo "Puede volver a reservar pasadas " . $t_reserva/3600 . " horas de
la hora de reserva escogida.<br /><br />"
    . " En cualquier otra duda, contacte con el administrador.<br /><br
/>";
    echo '<a class="button-fail button-fail-orange pure-button" href = "'
. $pathInclude . 'inicio.php">Regresar<br /></a>';
    mysqli_stmt_close($stmt);
    mysqli_close($conexion);
}

function reservaOkErrorHoraReservada($stmt, $conexion, $pathInclude){
    echo "Error. La franja horaria seleccionada ya ha sido reservada.<br
/>"
    ."Por favor, elija otra.<br /><br />" . mysqli_connect_error();
    echo '<a class="button-fail button-fail-orange pure-button" href = "'
. $pathInclude . 'inicio.php">Regresar<br /></a>';
    mysqli_stmt_close($stmt);
    mysqli_close($conexion);
}

function reservaOkErrorFecha($stmt, $conexion, $pathInclude){
    echo "Error. La fecha no ha sido introducida correctamente<br />" .
mysqli_connect_error();
    echo "Inténtelo de nuevo o contacte con el administrador";
    echo '<a href = "' . $pathInclude . 'inicio.php"><br />Regresar<a/>';
    mysqli_stmt_close($stmt);
    mysqli_close($conexion);
}
```

```
//<!--borrado_1.php y borradoConfirmacion.php-->

function borradoOkErrorEmailPass($stmt, $pathInclude){
    echo "Error. El usuario o contraseña no se encuentran en la BD<br />"
. mysqli_stmt_error($stmt);
    echo "Inténtelo de nuevo o contacte con el administrador<br /><br />";
    echo '<a class="button-fail button-fail-orange pure-button" href = "'
. $pathInclude . 'borrado.php">Regresar<br /></a>';
    mysqli_stmt_close($stmt);
}


function borradoOkErrorBorrado($stmt, $pathInclude){
    echo "Error.<br /> No se ha podido borrar o no había reserva
realizada.<br /><br />" . mysqli_stmt_error($stmt);
    echo '<a class="button-fail button-fail-orange pure-button" href = "'
. $pathInclude . 'borrado.php">Regresar<br /></a>';
    mysqli_stmt_close($stmt);
}

function borradoOkErrorDatos($pathInclude){
    echo "Error. El usuario o contraseña que se introdujeron no tienen el
formato adecuado.<br />";
    echo "Inténtelo de nuevo o contacte con el administrador<br /><br />";
    echo '<a class="button-fail button-fail-orange pure-button" href = "'
.$pathInclude . 'borrado.php">Regresar<br /></a>';
}

// Devuelve la hora actual en segundos
function horaActual(){

    $hora_actual = date("H:i:s");
    list($h, $m, $s) = explode(':',$hora_actual);
    $hora_actual_s = ($h * 3600) + ($m * 60) + $s;
    return $hora_actual_s;
}

function conectarDB(){

    $conexion = mysqli_connect(HOST,USER,PASSWORD,DATABASE);
    if(mysqli_connect_errno($conexion)){
        return null;
    }else{
        return $conexion;
    }
}

function borrarDirAlumno(){

    list($pathname, $pathnameVHD, $pathLib_remota, $myDir, $pathXilinx,
$pathReadUART) = asignacionRutas();

    if(file_exists($pathname .'\\out.txt')){
        if(!unlink($pathname .'\\out.txt')){
            echo "Error al borrar directorio 1";
        }elseif(file_exists($pathname .'\\errores.txt'))
            if(!unlink($pathname . '\\errores.txt')){
                echo "Error al borrar directorio 2";
            }elseif(file_exists($pathname .'\\out_impact.txt'))
                if(!unlink($pathname . '\\out_impact.txt')){
```

```
                    echo "Error al borrar directorio 3";
                }elseif(file_exists($pathname .'\\errores_impact.txt'))
                    if(!unlink($pathname . '\\errores_impact.txt')){
                        echo "Error al borrar directorio 4";
                    }
    }

    // Borramos posibles archivos dentro de la carpeta files creada al
alumno
    errorGeneracionBit($pathname, $pathnameVHD);
    // Borramos el directorio original del alumno en caso de que se
llegase a crear
    if(file_exists($pathname)){
        if(!rmdir($pathname)){
            echo "Error al borrar directorio";
        }
    }
}
```

II.2.16) style.css

```
* {
    -webkit-box-sizing: border-box;
    -moz-box-sizing: border-box;
    box-sizing: border-box;
   -o-box-sizing:border-box;

}

body{
    line-height: 1.7em;
    color: #7f8c8d;
    font-size: 13px;
}

h1,
h2,
h3,
h4,
h5,
h6,
label{
    color: #34495e;
}

.l-box{
    padding: 1em;
}

.is-center{
    text-align: center;
}
/*
 * Estilos de Menú
 */
.home-menu{
```

```
    padding: 0.5em;
    text-align: center;
    box-shadow: 0 20px 10px rgba(0,0,0, 0.10);
    background:#1F3D5C;
}

.pure-menu.pure-menu-fixed{
    border-bottom:none;
    z-index: 4;
}

.home-menu .pure-menu-heading{
    color: white;
    font-weight: 400;
    font-size: 120%;
}

.home-menu a{
    color: #33ADD6;
}

.home-menu a:hover,
.home-menu a:focus{
    background-color:#4D8F8F;
    border:none;
    color: #99D6EA;
}
/*
* Contenedores
*/
.cont-ppal{
    background: #1f8dd6;
    z-index: 1;
    overflow: hidden;
    width: 100%;
    height: 88%;
    top: 0;
    left: 0px;
    position: fixed !important;
}
.cont-ppal-upload_1{
    background: #1f8dd6;
    z-index: 1;
    overflow: hidden;
    width: 100%;
    height: 88%;
    top: 0;
    left: 0px;
    position: fixed !important;
}

.cont{
    width: 60%;
    height: 50%;
    margin: auto;
    position: absolute;
    top: 20px; left: 0; bottom: 0; right: 0;
    text-align: center;
    text-transform: uppercase;
}
```

```
.cont-calendar{
    width: 40%;
    height: 40%;
    margin: auto;
    position: absolute;
    top: 0px; left: 0; bottom: 0; right: 0;
    text-align: center;
    text-transform: none;
}

.cont-reserva{
    width: 40%;
    height: 70%;
    margin: auto;
    position: absolute;
    top: 0px; left: 0; bottom: 0; right: 0;
    text-align: center;
    text-transform:none;
}

.cont-upload{
    width: 80%;
    height: 70%;
    margin: auto;
    position: absolute;
    top: 0px; left: 0; bottom: 0; right: 0;
    text-align: center;
    text-transform:none;
}

.cont-upload_1{
    overflow-y: auto;
    width: 70%;
    height: 70%;
    margin: auto;
    position: absolute;
    top: 0px; left: 0; bottom: 0; right: 0;
    text-align: center;
    text-transform:none;
}
/*
* Contenedores de cabecera y centrales
*/
.cont-cabecera{
    font-size: 45px;
    font-weight: bolder;
    color: white;
    border: 12px solid white;
    padding: 0.5em 1em;
    font-weight: 100;
    border-radius: 8px;
    line-height: 1em;
}
.cont-cabecera-upload_1{
    font-size: 16px;
    font-weight: bold;
    color: white;
    border: 12px solid white;
    padding: 0.5em 1em;
```

```
    border-radius: 8px;
    line-height: 1em;
}
.cont-cabecera-calendar{
    font-size: 18px;
    font-weight: bold;
    color: black;
    border: 8px solid white;
    padding: 0.5em 1em;
    font-weight: normal;
    border-radius: 8px;
    line-height: 1em;
}
.cont-cabecera-reserva{
    font-size: 20px;
    font-weight: bold;
    color: black;
    border: 8px solid white;
    padding: 0.5em 1em;
    font-weight: bolder;
    border-radius: 8px;
    line-height: 1em;
}
.cont-cabecera-upload{
    font-size: 14px;
    font-weight: normal;
    color: black;
    border: 8px solid white;
    padding: 0.5em 1em;
    font-weight: bolder;
    border-radius: 8px;
    line-height: 1em;
}
/*
* Diseño de botones
*/
.button-success{
    color:black;
    border-radius: 50px;
    text-shadow: 0 1px 1px rgba(0, 0, 0, 0.2);
}
.button-success-green{
    border-radius: 50px;
    background: #00A300; // Color verde
}
.button-fail{
    color:black;
    border-radius: 50px;
    text-shadow: 0 1px 1px rgba(0, 0, 0, 0.2);
}
.button-fail-orange{
    background: #FFA500; // Color naranja
}
.button-fail-red{
    background: #F08080; // Color rojo
}
/*
* Botones Lab
*/
.radio-lab {
    opacity: 0;
```

```
    position: relative;
}

.radio-lab, .radio-lab-label{
    display: inline-block;
    vertical-align: middle;
    margin: 1px;
    cursor: pointer;
}
.radio-lab + .radio-lab-label:before{
    content: '';
    background: #fff;
    border: 4px solid #ddd;
    display: inline-block;
    vertical-align: middle;
    width: 18px;
    height: 18px;
    padding: 2px;
    margin-right: 7px;
    margin-top: 10px;
    margin-bottom: 5px;
    font-weight: bolder;
    text-align: center;
}

.radio-lab + .radio-custom-label:before{
    border-radius: 50%;
}

.radio-lab:checked + .radio-lab-label:before{
    content: "\f00c";
    font-family: "FontAwesome";
    font-size: 12px;
    font-weight: bolder;
    color: #009900;
}

.content-wrapper{
    position: absolute;
    top: 88%;
    width: 100%;
    min-height: 12%;
    z-index: 2;
    background: white;
}
.nueva{
 //   overflow-y: scroll;
}

.footer{
    background:#2b5ab8;
    color: #e0e0e0;
    font-weight: bolder;
}
/*
* Estilo para las páginas de interfazLab.php que muestran la interfaz
*/
.content-wrapper-lab-interfaz{
   // overflow-y: visible;
    position: absolute;
```

```
    top: 50%;
    width: 100%;
    min-height: 12%;
    z-index: 2;
    background: white;
}

.mostrar_consigna{
    position: inherit;
}

.cont-container-lab-prueba-29{
    background: #1f8dd6;
    z-index: 1;
    overflow: visible;
    width: 100%;
    height: 100%;
    top: 100px;
    bottom: 0px;
    left: 0px;
    position: fixed !important;
}

.cont-lab-prueba-29{
    width: 90%;
    height: 90%;
    margin: auto;
    position: absolute;
    top: -40px; left: 0; bottom: 0; right: 0;
    text-align: center;
    font-weight: bold;
    color: black;
    text-transform:none;
}
/*
 * Visualización para dispositivos móviles y tablets
 */
@media (min-width: 48em) {
    body{
        font-size: 16px;
    }

    .content{
        padding: 1em;
    }

    .home-menu{
        text-align: left;
    }
    .home-menu ul{
        float: right;
    }

    .splash{
        width: 50%;
        height: 50%;
    }

    .splash-head{
        font-size: 250%;
```

```
    }
}
/*
* Visualización para dispositivos con pantalla de gran tamaño
*/
@media (min-width: 78em){
    .splash-head{
        font-size: 300%;
    }
}
```

II.2.17) style_2.css

```
* {
    -webkit-box-sizing: border-box;
    -moz-box-sizing: border-box;
    box-sizing: border-box;
   -o-box-sizing:border-box;

}

body{
    line-height: 1.7em;
    color: #7f8c8d;
    font-size: 13px;
}

h1,
h2,
h3,
h4,
h5,
h6,
label{
    color: #34495e;
}

.l-box{
    padding: 1em;
}

.is-center{
    text-align: center;
}

/*
 * Estilos de Menú
 */

.home-menu{
    padding: 0.01em;
    text-align:center;
    box-shadow: 0 10px 10px rgba(0,0,0, 0.10);
    background:#1F3D5C;
```

```
}

.pure-menu.pure-menu-fixed{
    border-bottom:none;
    z-index: 4;
}

.home-menu .pure-menu-heading{
    color: white;
    font-weight: 600;
    font-size: 80%;
}

.home-menu a{
    color: #33ADD6;
}

.home-menu a:hover,
.home-menu a:focus{
    background-color:#4D8F8F;
    border:none;
    color: #99D6EA;
}
/*
* Contenedores
*/
.cont-ppal{
    background: #1f8dd6;
    z-index: 1;
    overflow: hidden;
    width: 100%;
    height: 88%;
    top: 0;
    left: 0px;
    position: fixed !important;
}
.cont-ppal-upload_1{
    background: #1f8dd6;
    z-index: 1;
    overflow: hidden;
    width: 100%;
    height: 88%;
    top: 0;
    left: 0px;
    position: fixed !important;
}

.cont{
    width: 60%;
    height: 50%;
    margin: auto;
    position: absolute;
    top: 20px; left: 0; bottom: 0; right: 0;
    text-align: center;
    text-transform: uppercase;
}

.cont-calendar{
    width: 40%;
    height: 40%;
```

```
    margin: auto;
    position: absolute;
    top: 10px; left: 0; bottom: 0; right: 0;
    text-align: center;
    text-transform: none;
}

.cont-reserva{
    width: 40%;
    height: 70%;
    margin: auto;
    position: absolute;
    top: 0px; left: 0; bottom: 0; right: 0;
    text-align: center;
    text-transform:none;
}

.cont-upload{
    width: 80%;
    height: 70%;
    margin: auto;
    position: absolute;
    top: 0px; left: 0; bottom: 0; right: 0;
    text-align: center;
    text-transform:none;
}

.cont-upload_1{
    overflow-y: auto;
    width: 70%;
    height: 70%;
    margin: auto;
    position: absolute;
    top: 0px; left: 0; bottom: 0; right: 0;
    text-align: center;
    text-transform:none;
}

/*
* Contenedores de cabecera y centrales
*/
.cont-cabecera{
    font-size: 45px;
    font-weight: bolder;
    color: white;
    border: 12px solid white;
    padding: 0.5em 1em;
    font-weight: 100;
    border-radius: 8px;
    line-height: 1em;
}
.cont-cabecera-upload_1{
    font-size: 16px;
    font-weight: bold;
    color: white;
    border: 12px solid white;
    padding: 0.5em 1em;
    border-radius: 8px;
    line-height: 1em;
}
```

```
.cont-cabecera-calendar{
    font-size: 20px;
    font-weight: bold;
    color: black;
    border: 8px solid white;
    padding: 0.5em 1em;
    font-weight: normal;
    border-radius: 8px;
    line-height: 1em;
}
.cont-cabecera-reserva{
    font-size: 20px;
    font-weight: bold;
    color: black;
    border: 8px solid white;
    padding: 0.5em 1em;
    font-weight: bolder;
    border-radius: 8px;
    line-height: 1em;
}
.cont-cabecera-upload{
    font-size: 14px;
    font-weight: normal;
    color: black;
    border: 8px solid white;
    padding: 0.5em 1em;
    font-weight: bolder;
    border-radius: 8px;
    line-height: 1em;
}

/*
* Diseño de botones
*/
.button-success{
    color:black;
    border-radius: 50px;
    text-shadow: 0 1px 1px rgba(0, 0, 0, 0.2);
}
.button-success-green{
    border-radius: 50px;
    background: #00A300; // Color verde
}
.button-fail{
    color:black;
    border-radius: 50px;
    text-shadow: 0 1px 1px rgba(0, 0, 0, 0.2);
}
.button-fail-orange{
    background: #FFA500; // Color naranja
}
.button-fail-red{
    background: #F08080; // Color rojo
}

/*
* Botones Lab
*/
.radio-lab {
    opacity: 0;
    position: relative;
```

```
}

.radio-lab, .radio-lab-label{
    display: inline-block;
    vertical-align: middle;
    margin: 1px;
    cursor: pointer;
}
.radio-lab + .radio-lab-label:before{
    content: '';
    background: #fff;
    border: 4px solid #ddd;
    display: inline-block;
    vertical-align: middle;
    width: 16px;
    height: 18px;
    padding: 2px;
    margin-right: 7px;
    margin-top: 10px;
    margin-bottom: 5px;
    font-weight: bolder;
    text-align: center;
}

.radio-lab + .radio-custom-label:before{
    border-radius: 50%;
}

.radio-lab:checked + .radio-lab-label:before{
    content: "\f00c";
    font-family: "FontAwesome";
    font-size: 12px;
    font-weight: bolder;
    color: #009900;
}

.content-wrapper{
    position: absolute;
    top: 88%;
    width: 100%;
    min-height: 12%;
    z-index: 2;
    background: white;
}

.footer{
    background:#2b5ab8;
    color: #e0e0e0;
    font-weight: bolder;
}

.slider{
    display: inline;
}

.mi_consigna{
    position: relative;
}
.nueva{
    position:relative;
```

```
    overflow: scroll;
    top: -50px; //aqui para subir los botones más hacia el iframe
}
.mostrar_consigna{
    position: inherit;
 //   text-align: right;
}

/*
* Estilo para las páginas de interfazLab.php que muestran la interfaz
*/
.content-wrapper-lab-interfaz{
    overflow-y: visible;
    position: absolute;
    top: 50%;
    width: 100%;
    min-height: 12%;
    z-index: 2;
    background: white;
}



.cont-container-lab-prueba-29{
    background: #1f8dd6;
    z-index: 1;
    overflow: visible;
    width: 100%;
    height: 100%;
    top: 100px;
    bottom: 0px;
    left: 0px;
    position: fixed !important;
}

.cont-lab-prueba-29{
    width: 100%;
    height: 90%;
    margin: auto;
    position: absolute;
    top: -40px; left: 0; bottom: 0; right: 0;
    text-align: center;
    font-weight: bold;
    color: black;
    text-transform:none;
}

/*
 * Visualización para dispositivos móviles y tablets
 */
@media (min-width: 48em) {
    body{
        font-size: 16px;
    }

    .content{
        padding: 1em;
    }

    .home-menu{
```

```
        text-align: left;
    }
    .home-menu ul{
        float: right;
    }

    .splash{
        width: 50%;
        height: 50%;
    }

    .splash-head{
        font-size: 250%;
    }
}
/*
* Visualización para dispositivos con pantalla de gran tamaño
*/
@media (min-width: 78em){
    .splash-head{
        font-size: 300%;
    }
}
```

II.2.18) UART_2.cpp

```
#include <windows.h>
#include <tchar.h>
#include <stdio.h>

void PrintCommState(DCB dcb)
{
    //  Print some of the DCB structure values
    _tprintf( TEXT("\nBaudRate = %d, ByteSize = %d, Parity = %d, StopBits
= %d\n"),
              dcb.BaudRate,
              dcb.ByteSize,
              dcb.Parity,
              dcb.StopBits );
}

int _tmain(int argc, _TCHAR* argv[])
{
    DCB dcb;
   HANDLE hCom;
   BOOL fSuccess;
   TCHAR *pcCommPort = TEXT("COM8"); //  Most systems have a COM1 port

   //  Open a handle to the specified com port.
   hCom = CreateFile( pcCommPort,
                      GENERIC_READ,
                      FILE_SHARE_READ,      //  must be opened with
exclusive-access
                      NULL,   //  default security attributes
                      OPEN_EXISTING, //  must use OPEN_EXISTING
                      0,      //  not overlapped I/O
                      NULL ); //  hTemplate must be NULL for comm devices
```

```
    if (hCom == INVALID_HANDLE_VALUE)
    {
        //  Handle the error.
        //printf ("CreateFile failed with error %d.\n", GetLastError());
        return (-1);
    }

    //  Initialize the DCB structure.
    SecureZeroMemory(&dcb, sizeof(DCB));
    dcb.DCBlength = sizeof(DCB);

    //  Build on the current configuration by first retrieving all current
    //  settings.
    fSuccess = GetCommState(hCom, &dcb);

    if (!fSuccess)
    {
       //  Handle the error.
       //printf ("GetCommState failed with error %d.\n", GetLastError());
       return (-1);
    }

    //PrintCommState(dcb);       //  Output to console

    //  Fill in some DCB values and set the com state:
    //  19,200 bps, 8 data bits, no parity, and 1 stop bit.
    dcb.BaudRate = CBR_19200;     //  baud rate
    dcb.ByteSize = 8;             //  data bits
    dcb.Parity   = NOPARITY;      //  parity bit
    dcb.StopBits = ONESTOPBIT;    //  stop bit

    fSuccess = SetCommState(hCom, &dcb);

    if (!fSuccess)
    {
       //  Handle the error.
       //printf ("SetCommState failed with error %d.\n", GetLastError());
       return (-1);
    }

    //  Get the comm config again.
    fSuccess = GetCommState(hCom, &dcb);

    if (!fSuccess)
    {
       //  Handle the error.
       //printf ("GetCommState failed with error %d.\n", GetLastError());
       return (-1);
    }

    //PrintCommState(dcb);       //  Output to console

    // tprintf (TEXT("Serial port %s successfully reconfigured.\n"),
pcCommPort);

     int retVal=0;
     BYTE Byte=0;
     DWORD dwBytesTransferred=0;
     DWORD dwCommModemStatus=0;
```

```
    SetCommMask (hCom, EV_RXCHAR | EV_ERR); //receive character event
    WaitCommEvent (hCom, &dwCommModemStatus, 0); //wait for character
    if (dwCommModemStatus & EV_RXCHAR)
        ReadFile (hCom, &Byte, 1, &dwBytesTransferred, 0); //read 1
    else if (dwCommModemStatus & EV_ERR)
        retVal = -1;
    retVal = Byte;

    _tprintf (TEXT("%d"), retVal);

    CloseHandle(hCom);

    return retVal;

}
```

II.3 Desarrollo en servidor web: insertarBD

II.3.1) inicioBD.php

```
<!-- inicioBD.php -->
<?php include("cabeceraBD.inc");
      include("funcionesBD.php"); ?>

<div class="cont-ppal">
    <div class="cont">
        <h1 class="cont-cabecera"> Gestión <br /> Base de Datos <br /><br
/> Laboratorio <br/> Sistemas <br/> de <br/> Control</h1>
    </div>
</div>
<div class="content-wrapper">
    <div class="content">

<?php include("pieBD.inc");?>
```

II.3.2) cabeceraBD.inc

```
<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <link rel="stylesheet" href="http://netdna.bootstrapcdn.com/font-
awesome/4.0.3/css/font-awesome.css">
    <link rel="stylesheet"
href="http://yui.yahooapis.com/pure/0.6.0/grids-responsive-min.css">
    <link rel="stylesheet" href="http://yui.yahooapis.com/pure/0.6.0/pure-
min.css">
    <link rel="stylesheet" href="/insertarBD/style.css">

    <title>Gestión Base de Datos</title>

</head>
<body>
    <div class="header">
        <div class="home-menu pure-menu pure-menu-horizontal pure-menu-
fixed">
            <a class="pure-menu-heading"
href="/insertarBD/inicioBD.php">Inicio</a>
            <a class="pure-menu-heading"
href="/insertarBD/insertar_BD.php">Insertar Alumnos BD</a>
            <a class="pure-menu-heading"
href="/insertarBD/borrar_BD.php">Borrar Alumnos BD</a>
        </div>
    </div>
```

II.3.3) pieBD.inc

```
    <div class="footer l-box is-center">
        Human Computer Technology Laboratory | Escuela Politécnica
Superior | Universidad Autónoma de Madrid
    </div>
</div>
</div>

</body>
</html>
```

II.3.4) insertar_BD.php

```
<!-- insertar_BD.php -->

<?php session_start();
      include("cabeceraBD.inc");
      include("funcionesBD.php");?>

<div class="cont-ppal">
    <div class="cont-upload">
        <div class="cont-cabecera-upload">

            <!--Subimos el fichero y enviamos los datos de la sesión-->

            <div class="pure-form ">
                Suba su fichero .csv generado en Excel con el nombre de
usuario (email institucional de la UAM) <br />
                                    y la contraseña que le quiere asignar.<br
/><br />
                    Deberá ser una fila por cada (usuario + contraseña) y
cada uno <br />
                                    de esos datos en una columna diferente.
                <form action = "insertar_BD_1.php" method = "post" enctype
= "multipart/form-data">
                    <label for = "archivo"> Archivos .csv a cargar
</label>
                    <i class="fa fa-upload fa-2x"></i>
                    <input type="file" name="archivo" id="archivo"/><br
/><br />
                    <button type ="submit" class="button-success button-
success-green pure-button" ><b>Enviar</b></button><br /><br />
                </form>
            </div>
        </div>
    </div>
</div>
 <div class="content-wrapper">
    <div class="content">

<?php include("pieBD.inc"); ?>
```

II.3.5) insertar_BD_1.php

```
<!-- insertar BD 1.php -->

<?php session_start();
      include("cabeceraBD.inc");
      include("funcionesBD.php");?>

<div class="cont-ppal-upload_1">
    <div class="cont-upload_1">
        <div class="cont-cabecera-upload_1">
<?php
       // Comprobamos si el fichero que ha subido es válido para nuestra
interfaz
        $val_manejo_archivo = manejoArchivo($pathIncludeFile);

            //Si ha habido algún error con el fichero, regreso a cargar
uno nuevo
            if($val_manejo_archivo === 0){
                ?><form action ="insertar_BD.php" method = "post" enctype
= "multipart/form-data"><br /><br />
                    <button type ="submit" class="button-fail button-fail-
orange pure-button"><b>Regresar</b></button>
                </form><?php
            }else{
            //Si no hay errores con el fichero subido, terminamos y
regreso al inicio
                echo "Se ha actualizado la Base de Datos correctamente.<br
/><br />";
                echo '<a class="button-fail button-success-green pure-
button" href = "' .
                    $pathInclude . 'inicioBD.php">Regresar<br /></a>';

            }?>
        </div>
    </div>
</div>
 <div class="content-wrapper">
    <div class="content">

<?php include("pieBD.inc"); ?>
```

II.3.6) borrar_BD.php

```
<!-- borrar_BD.php -->

<?php session_start();
      include("cabeceraBD.inc");
      include("funcionesBD.php");?>

<div class="cont-ppal-upload_1">
    <div class="cont-upload_1">
        <div class="cont-cabecera-upload_1">

            <div class="pure-form ">
                        ¿Desea borrar todas las entradas de la Base de
Datos?
            <form action = "borrar_BD_1.php" method = "post" enctype =
"multipart/form-data">
```

```
            <input id= "radio-1" class="radio-lab" name="borrado"
type="radio" value = "1">
            <label for= "radio-1" class="radio-lab-label">Sí</label>
            <input id= "radio-2" class="radio-lab" name="borrado"
type="radio" value = "2" checked>
            <label for= "radio-2" class="radio-lab-label">No</label><br />
<br />

            <button type ="submit" class="button-success button-success-
green pure-button" ><b>Enviar</b></button><br /><br />
                                    Esta operación no es reversible.<br />
                Si desea rellenar de nuevo la Base de Datos deberá cargar
un nuevo fichero .csv<br />
            </div>
        </div>
    </div>
</div>
 <div class="content-wrapper">
    <div class="content">

<?php include("pieBD.inc"); ?>
```

II.3.7) borrar_BD_1.php

```
<!-- borrar_BD_1.php -->

<?php session_start();
      include("cabeceraBD.inc");
      include("funcionesBD.php");?>

<div class="cont-ppal-upload_1">
    <div class="cont-upload_1">
        <div class="cont-cabecera-upload_1">
<?php
    if($_POST['borrado'] == 1){
    // Se ha seleccionado que no quiere borrar los datos de la tabla
usuarios
        $borrado = 1;
    }elseif($_POST['borrado'] == 2){
        // Sí se quieren borrar los datos de dicha tabla
        $borrado = 0;
    }

    if($borrado == 1){

        $conexion = conectarDB();
        if($conexion == null){
            return 0;
        }
        $consulta = "SELECT id_usuario FROM usuarios";
        if($stmt = mysqli_prepare($conexion, $consulta)){
            if(!mysqli_stmt_execute($stmt)){
                return 0;
            }else{
                // Almacenamos el resultado
                mysqli_stmt_store_result($stmt);
                // Obtenemos los resultados y los almacenamos en una
variable
```

```
                mysqli_stmt_bind_result($stmt, $IDs);
                // Mientras haya resultados, ejecutamos un borrado por
cada ID de la tabla
                while(mysqli_stmt_fetch($stmt)){
                    $sentencia = "DELETE FROM `hctlab`.`usuarios` WHERE
`id_usuario` =?";
                    if($stmt_1 = mysqli_prepare($conexion, $sentencia)){
                        mysqli_stmt_bind_param($stmt_1, 's', $IDs);
                        if(!mysqli_stmt_execute($stmt_1)){
                            echo "Error en el borrado de los datos.<br />
<br />";
                            echo '<a class="button-fail button-fail-orange
pure-button" href = "' .
                                    $pathInclude .
'inicioBD.php">Regresar<br /></a>';
                            mysqli_stmt_close($stmt);
                            mysqli_close($conexion);
                            return 0;
                        }else{
                            mysqli_stmt_close($stmt_1);
                        }
                    }else{
                        echo "Error en el borrado de los datos.<br /> <br
/>";
                        echo '<a class="button-fail button-fail-orange
pure-button" href = "' .
                                $pathInclude . 'inicioBD.php">Regresar<br
/></a>';
                        mysqli_stmt_close($stmt);
                        mysqli_close($conexion);
                        return 0;
                    }
                }
                mysqli_stmt_close($stmt);
                echo "Se han borrado con éxito todos los datos de los
usuarios.<br /> <br />";
            }
        }
    }else{
        echo "Ha decidido no borrar su reserva.<br /> <br />";
        echo '<a class="button-success button-success-green pure-button" '
            . 'href = "' . $pathInclude . 'inicioBD.php">Página
principal<br /></a></h1>';
    }?>
        </div>
    </div>
</div>
 <div class="content-wrapper">
    <div class="content">

<?php include("pieBD.inc"); ?>
```

II.3.8) funciones.php

```
<!-- funciones.php -->
<?php
// Directorio donde se encuentran alojadas las páginas php
global $pathInclude;
       $pathInclude =(string)"/insertarBD/";

global $pathIncludeFile;
       $pathIncludeFile ="C:/Servidor/insertarBD/actualizar/";
// Definición de constantes para la BD
// Alojamiento al que nos conectamos
define("HOST", "localhost");
// Nombre de usuario de la Base de Datos
define("USER", "epsuam");
// Contraseña de la Base de Datos
define("PASSWORD", "xxxxxxxx");
// Nombre de la Base de Datos utilizada
define("DATABASE", "hctlab");

function manejoArchivo($pathIncludeFile){

    // Obtenemos únicamente el nombre del fichero, sin la ruta
    $nombre = basename($_FILES['archivo']['name']);

    // Solo permitimos subir archivo .bit
    $extension_permitida = array('csv');
    // Extensiones extra no permitidas
    $extensiones_invalidas = array('php','php3','php4','phtml','exe');
    // Caracteres permitidos en el nombre del fichero
    $caracteres_validos = '/[A-Za-z0-9]/';
    // Nos quedamos con la extensión del archivo
    $extension = end(explode('.', basename($_FILES['archivo']['name'])));
    // Comprobamos si se encuentra la extensión permitida
    $extension_ok = in_array($extension, $extension_permitida);

    $nombre_unico = basename($_FILES['archivo']['name'], ".csv");

        // Comprobamos que se ha enviado un archivo y que no han sido
varios archivos
        if($_FILES['archivo']['size'] == 0){
            echo 'No se ha seleccionado archivo <br />';
            return 0;
        }elseif(in_array(end(explode('.',
basename($_FILES['archivo']['name']))), $extensiones_invalidas)){
            echo "Extensión no permitida. <br /> <br />";
            return 0;
        }elseif(preg_match($caracteres_validos,
basename($_FILES['archivo']['name']))!= 1){
                echo "Nombre de archivo inválido. Renómbrelo de otra
manera. <br /><br />";
                echo "pase";
                return 0;
            if(preg_match("/exe/", basename($_FILES['archivo']['name'] ==
1))){
                echo "Nombre de archivo inválido. Renómbrelo de otra
manera. <br /><br />";
                echo "pase";
                return 0;
            }
```

```
        }elseif( $_FILES['archivo']['error'] > 0){

                switch($_FILES['archivo']['error']){
                    case 1:     // UPLOAD_ERR_INI_SIZE
                    case 2:     // UPLOAD_ERR_FORM_SIZE
                        echo 'Tamaño máximo permitido. Suba de nuevo un
archivo válido';
                        break;
                    case 3:     // UPLOAD_ERR_PARTIAL
                        echo 'Archivo subido parcialmente. Suba de nuevo un
archivo válido';
                        break;
                    case 4:     // UPLOAD_ERR_NO_FILE
                        echo 'No se ha subido archivo. Suba uno';
                        break;
                    case 6:     // UPLOAD_ERR_NO_TMP_DIR
                        echo 'Directorio temporal no encontrado Suba de nuevo
un archivo';
                        break;
                    case 7:     // UPLOAD_ERR_CANT_WRITE
                        echo 'Error al escribir en disco. Suba de nuevo un
archivo';
                        break;
                    case 8:     // UPLOAD_ERR_EXTENSION
                        echo 'Una extensión de PHP paró la subida. Suba de
nuevo un archivo';
                        break;
                    default :
                        echo 'Error en archivo. Suba uno de nuevo';
                        break;
                }
                return 0;
        }elseif($extension_ok){

                $res_procesado = procesadoCSV($pathIncludeFile, $nombre_unico,
$extension_permitida);
                if($res_procesado === 0){
                    return 0;
                }else{
                    return 1;
                }

        }else{
                 echo 'Archivo inválido. Seleccione archivo <b>.csv</b>
correcto';
                 return 0;
        }

    return 1;
}

function procesadoCSV($pathIncludeFile, $nombre_unico,
$extension permitida){

    $nombre_tmp = $_FILES['archivo']['tmp_name'];
    //Ruta y nombre del fichero almacenado
    $pathFile = $pathIncludeFile . $nombre_unico . '.' .
$extension_permitida[0];

    if(file_exists($pathFile)){
```

```
        if(!unlink($pathFile)){
           echo "Error al borrar fichero .csv <br />";
        }
        if(!rmdir($pathIncludeFile)){
            echo "Error al borrar el directorio del .csv almacenado. <br
/>";
        }
    }elseif(file_exists($pathIncludeFile)){
         if(!rmdir($pathIncludeFile)){
            echo "Error al borrar el directorio del .csv almacenado. <br
/>";
        }
    }

    if(!mkdir($pathIncludeFile, 0777, true)){
        echo "Error al crear la carpeta para guardar el fichero <br />";
        return 0;
    }

    if(!move_uploaded_file($nombre_tmp, $pathFile)){
        echo "No ha sido posible copiar el fichero .csv<br /> Inténtelo de
nuevo";
        return 0;
    }

        $file = fopen($pathFile, "r");
        if(!$file){
            echo "Ha habido un error en la apertura del fichero. <br />";
            return 0;
        }else{
            while(($data = fgetcsv($file, 0, ";")) !== false){
                // Creamos un array por cada par de datos leídos
                $datos = [];
                $num = count($data);
                // Leemos de 2 en 2 datos
                for($c=0; $c < $num; $c+=2){
                    for($i = 0; $i < 2; $i++){
                        $datos[] = $data[$c + $i];
                    }
                }
                $res = procesadoDatos($datos);
                if($res === 0){
                    echo "Error en la inserción en la Base de Datos";
                    fclose($file);
                    unlink($pathFile);
                    rmdir($pathIncludeFile);
                    return 0;

                }else{
                    continue;
                }
            }
        }
    fclose($file);
    // Borramos el fichero y su carpeta creada
    unlink($pathFile);
    rmdir($pathIncludeFile);

    return 1;
}
```

```php
// Función para insertar los datos de los alumnos
function  procesadoDatos($datos){

    $conexion = conectarDB();
    if ($conexion == null){
        return 0;
    }
    $hash = password_hash($datos[1], PASSWORD_BCRYPT);

    $sentencia = "INSERT INTO hctlab.usuarios(`email`,`hash`,`acceso`)
VALUES (?,?,0)";
    if($stmt = mysqli_prepare($conexion, $sentencia)){
        // Agregamos las variables como parámetros a la sentencia
preparada
        mysqli_stmt_bind_param($stmt, 'ss', $datos[0], $hash);
        if(!mysqli_stmt_execute($stmt)){
            echo "Ya existen registros de usuarios. <br/>";
            echo "Borre la actual base de datos e introduzca los nuevos
datos.<br />";
            return 0;
        }else{
            mysqli_stmt_close($stmt);
            mysqli_close($conexion);
            return 1;
        }
    }else{
        mysqli_stmt_close($stmt);
        mysqli_close($conexion);
        return 0;
    }
}

function conectarDB(){

    $conexion = mysqli_connect(HOST,USER,PASSWORD,DATABASE);

    if(mysqli_connect_errno($conexion)){
        return null;
    }else{
        return $conexion;
    }
}
```

II.3.9) style.css

```css
* {
    -webkit-box-sizing: border-box;
    -moz-box-sizing: border-box;
    box-sizing: border-box;
   -o-box-sizing:border-box;

}

body{
    line-height: 1.7em;
    color: #7f8c8d;
    font-size: 13px;
}
```

```
h1,
h2,
h3,
h4,
h5,
h6,
label{
    color: #34495e;
}

.l-box{
    padding: 1em;
}

.is-center{
    text-align: center;
}
/*
 * Estilos de Menú
 */
.home-menu{
    padding: 0.5em;
    text-align: center;
    box-shadow: 0 20px 10px rgba(0,0,0, 0.10);
    background:#1F3D5C;
}

.pure-menu.pure-menu-fixed{
    border-bottom:none;
    z-index: 4;
}

.home-menu .pure-menu-heading{
    color: white;
    font-weight: 400;
    font-size: 120%;
}

.home-menu a{
    color: #33ADD6;
}

.home-menu a:hover,
.home-menu a:focus{
    background-color:#4D8F8F;
    border:none;
    color: #99D6EA;
}
/*
* Contenedores
*/
.cont-ppal{
    background: #1f8dd6;
    z-index: 1;
    overflow: hidden;
    width: 100%;
    height: 88%;
    top: 0;
```

```
    left: 0px;
    position: fixed !important;
}
.cont-ppal-upload_1{
    background: #1f8dd6;
    z-index: 1;
    overflow: hidden;
    width: 100%;
    height: 88%;
    top: 0;
    left: 0px;
    position: fixed !important;
}

.cont{
    width: 60%;
    height: 50%;
    margin: auto;
    position: absolute;
    top: 20px; left: 0; bottom: 0; right: 0;
    text-align: center;
    text-transform: uppercase;
}

.cont-upload{
    width: 80%;
    height: 70%;
    margin: auto;
    position: absolute;
    top: 0px; left: 0; bottom: 0; right: 0;
    text-align: center;
    text-transform:none;
}

.cont-upload_1{
    overflow-y: auto;
    width: 70%;
    height: 70%;
    margin: auto;
    position: absolute;
    top: 0px; left: 0; bottom: 0; right: 0;
    text-align: center;
    text-transform:none;
}
/*
* Contenedores de cabecera y centrales
*/
.cont-cabecera{
    font-size: 45px;
    font-weight: bolder;
    color: white;
    border: 12px solid white;
    padding: 0.5em 1em;
    font-weight: 100;
    border-radius: 8px;
    line-height: 1em;
}
.cont-cabecera-upload_1{
    font-size: 16px;
    font-weight: bold;
    color: white;
```

```
    border: 12px solid white;
    padding: 0.5em 1em;
    border-radius: 8px;
    line-height: 1em;
}

.cont-cabecera-upload{
    font-size: 14px;
    font-weight: normal;
    color: black;
    border: 8px solid white;
    padding: 0.5em 1em;
    font-weight: bolder;
    border-radius: 8px;
    line-height: 1em;
}

/*
* Diseño de botones
*/
.button-success{
    color:black;
    border-radius: 50px;
    text-shadow: 0 1px 1px rgba(0, 0, 0, 0.2);
}
.button-success-green{
    border-radius: 50px;
    background: #00A300; // Color verde
}
.button-fail{
    color:black;
    border-radius: 50px;
    text-shadow: 0 1px 1px rgba(0, 0, 0, 0.2);
}
.button-fail-orange{
    background: #FFA500; // Color naranja
}
.button-fail-red{
    background: #F08080; // Color rojo
}
/*
* Botones Lab
*/
.radio-lab {
    opacity: 0;
    position: relative;
}

.radio-lab, .radio-lab-label{
    display: inline-block;
    vertical-align: middle;
    margin: 1px;
    cursor: pointer;
}
.radio-lab + .radio-lab-label:before{
    content: '';
    background: #fff;
    border: 4px solid #ddd;
    display: inline-block;
    vertical-align: middle;
```

```
    width: 18px;
    height: 18px;
    padding: 2px;
    margin-right: 7px;
    margin-top: 10px;
    margin-bottom: 5px;
    font-weight: bolder;
    text-align: center;
}

.radio-lab + .radio-custom-label:before{
    border-radius: 50%;
}

.radio-lab:checked + .radio-lab-label:before{
    content: "\f00c";
    font-family: "FontAwesome";
    font-size: 12px;
    font-weight: bolder;
    color: #009900;
}

.content-wrapper{
    position: absolute;
    top: 88%;
    width: 100%;
    min-height: 12%;
    z-index: 2;
    background: white;
}

.footer{
    background:#2b5ab8;
    color: #e0e0e0;
    font-weight: bolder;
}

/*
 * Visualización para dispositivos móviles y tablets
 */
@media (min-width: 48em) {
    body{
        font-size: 16px;
    }

    .content{
        padding: 1em;
    }

    .home-menu{
        text-align: left;
    }
    .home-menu ul{
        float: right;
    }

    .splash{
        width: 50%;
        height: 50%;
    }
```

```
    .splash-head{
        font-size: 250%;
    }
}
/*
* Visualización para dispositivos con pantalla de gran tamaño
*/
@media (min-width: 78em){
    .splash-head{
        font-size: 300%;
    }
}
```

```
position: absolute;
    top: 88%;
    width: 100%;
    min-height: 12%;
    z-index: 2;
    background: white;
}

.footer{
    background:#2b5ab8;
    color: #e0e0e0;
    font-weight: bolder;
}

/*
 * Visualización para dispositivos móviles y tablets
 */
@media (min-width: 48em) {
    body{
        font-size: 16px;
    }

    .content{
        padding: 1em;
    }

    .home-menu{
        text-align: left;
    }
    .home-menu ul{
        float: right;
    }

    .splash{
        width: 50%;
        height: 50%;
    }

    .splash-head{
        font-size: 250%;
    }
}
```

```
/*
* Visualización para dispositivos con pantalla de gran tamaño
*/
@media (min-width: 78em){
    .splash-head{
        font-size: 300%;
    }
}
```